

Pretty Good Privacy (PGP) & GNU Privacy Guard (GPG): Just Enough Training to Make You Dangerous

Joe St Sauver, Ph.D.
M3AAWG Senior Technical Advisor
Distributed System Scientist
Farsight Security, Inc.

M3AAWG 34, Dublin, Ireland
Monday, June 8th, 2015, 15:00-17:00

<https://www.stsauver.com/joe/pgp-tutorial/>

0. Introduction

Obligatory Screen: Eligibility For Strong Encryption

- This is not legal advice (for that, please contact your attorney), however please note that some people are NOT ALLOWED to use strong encryption under prevailing laws.
- **By continuing with this training, you certify that you are NOT:**
 - a citizen, national, or resident of a country barred from access to strong encryption by the U.S. or other countries, including but not limited to persons from the Crimea region of the Ukraine, Cuba, Iran, North Korea, Sudan, or Syria;
 - nor are you a "Specially Designated National" (see <http://www.treasury.gov/resource-center/sanctions/SDN-List/Pages/default.aspx>), nor a person (or representative of a company) that is subject to any other US or other sanctions program or restriction.
- If you are subject to any such prohibition or restriction, you must NOT participate in today's encryption training.

Disclaimer

- While all due care was used in preparing the content of this training, we cannot ensure that you will not inadvertently make a mistake, or encounter a vulnerability while using PGP/GPG.
- Given that you cannot "unring the bell once it has been rung," and given that some potential "losses of confidentiality" may have grave or even catastrophic consequences, please remember that:
 - you should not use PGP/GPG for "life/safety-critical" purposes
 - today's training is provided on a "best efforts," as-is, where-is basis, with all evident and/or latent faults/flaws
 - should you decide to use and rely on PGP/GPG, the decision to do so is your own and at your own risk; we disclaim all responsibility for any impacts associated with the use, misuse, or abuse of PGP/GPG by anyone here today or using this talk.
- If you do not agree with these terms, please do not rely on the information in this talk, and please do not stay for this training.

Our Goal Today: 'Basic PGP/GPG Functional Literacy'

- **Background**
 - understanding why you might need or want PGP/GPG
 - understanding criticisms/limitations of PGP/GPG
 - understanding (in general) how PGP/GPG works
- **Installing and Configuring PGP/GPG (Mac [and Linux], and MS Windows)**
- **Creating and Managing Keys**
 - creating and publishing your own keypair
 - getting other peoples' PGP/GPG keys
- **Digitally Signing/Validating Messages**
 - sending a *signed message*
 - validating a *signed message* you've received
- **Encrypting/Decrypting Messages**
 - sending an *encrypted message*
 - decrypting an *encrypted message* you've received
- **Signing PGP/GPG Keys**
- **Convenience Tools: Enigmail and Mailvelope**
- **Optional Key Signing Party**

I. Why Bother Learning To Use PGP/GPG?

All Email Is Private, Right?

- You might expect your email to be private, exchanged just between you and your intended correspondent(s). However, many people may potentially be in a position to look at it:
 - Network engineers/system administrators with privileged access
 - Supervisors/managers with administrative power to obtain access
 - Hardware techs with physical access when fixing hardware issues
 - Advertising/automatic marketing systems (at least on some free ad-sponsored email services)
 - Family members with access to automatically-logged in mail clients
 - Hacker/crackers with unauthorized access due to bugs/exploits
 - Civil litigants engaged in compulsory pre-trial discovery during a lawsuit
 - Law enforcement officers (LEOs) with a warrant or other paperwork (or exigent circumstances)
 - Members of the intelligence community (foreign or domestic)
 - Customs officers at international borders
 - FOIA requestors (for government officials), etc.

"But Joe! I've Got Nothing to Hide!"

- And that's terrific. Most folks don't. But sometimes **others** may want to share something with you in confidence – wouldn't it be nice if they could?
- Maybe you're trying to talk about spam or malware, and just need a way to avoid having your messages automatically filtered when your message contains malware samples or spam samples?
- In other cases, perhaps you need to know how to use PGP/GPG simply because a group you want to work with may require you to be able to do so, whether **you** perceive a need for it or not.
- Another possibility: maybe you'd like to **digitally sign** your email, so that there's no question that it came from you, and hasn't been tampered with since it was sent?
- Bottom line, there are many, many legitimate reasons why people might need or want to use PGP/GPG

"But What About PGP/GPG and the Bad Guys???"

- It's true – the bad guys are interested in PGP/GPG, too. Some really unsavory types (terrorists, pedophiles, organized crime members, etc.) would love to keep their messages away from the authorities. Those bad guys dream about the privacy that PGP/GPG provides (and they likely already use it).
- Sadly, there's no way that I know of to keep the bad guys from using PGP while still allowing everyone else to use it including:
 - law-abiding international businessmen
 - peaceful religious or political dissidents
 - investigative journalists, and
 - ordinary privacy-minded men and women.
- Criminal misuse can and *should* be punished, but lawful use by everyone else shouldn't be precluded.
- Not everyone agrees...

Comments From UK Prime Minister Cameron

- **[UK Prime Minister David] Cameron also said it must be possible for the government to read any form of communication, regardless of whether it is encrypted or not. He said there should be no means of communication which "we cannot read".**

"I think we cannot allow modern forms of communication to be exempt from the ability, in extremis, with a warrant signed by the home secretary, to be exempt from being listened to," he argued. He added that if he is prime minister after the next election he will "legislate accordingly" to increase state surveillance of the internet.

<http://www.wired.co.uk/news/archive/2015-01/13/david-cameron-snoopers-charter> [emphasis added][and yes, Cameron was re-elected, on May 7th, 2015]

- **"Cameron's encryption ban is 'infeasible', says Parliamentary office"**

<http://www.wired.co.uk/news/archive/2015-03/10/david-cameron-encryption-ban-impossible>

The UN Weighs In, Urging Protection for Encryption

- "59. **States should promote strong encryption** and anonymity. National laws should recognize that individuals are free to protect the privacy of their digital communications by using encryption technology and tools that allow anonymity online. Legislation and regulations protecting human rights defenders and journalists should also include provisions enabling access and providing support to use the technologies to secure their communications.
- 60. **States should not restrict encryption** and anonymity, which facilitate and often enable the rights to freedom of opinion and expression. Blanket prohibitions fail to be necessary and proportionate. States should avoid all measures that weaken the security that individuals may enjoy online, such as backdoors, weak encryption standards and key escrows. In addition, States should refrain from making the identification of users a condition for access to digital communications and online services and requiring SIM card registration for mobile users. Corporate actors should likewise consider their own policies that restrict encryption and anonymity (including through the use of pseudonyms). [continues]" "Report on encryption, anonymity, and the human rights framework," http://www.ohchr.org/EN/HRBodies/HRC/RegularSessions/Session29/Documents/A.HRC.29.32_AEV.doc [May 22nd, 2015, emphasis added]

II. PGP/GPG: "Too Hard To Learn To Use?"

PGP/GPG: A "Daunting" Crypto Option?





- PGP/GNU PrivacyGuard has often been viewed as technically daunting, even spawning academic studies such as the famous:
 - **"Why Johnny Can't Encrypt"** (1999), http://www.eecs.berkeley.edu/~tygar/papers/Why_Johnny_Cant_Encrypt/USENIX.pdf




"Our twelve test participants were generally educated and experienced at using email, yet **only one third of them were able to use PGP 5.0 to correctly sign and encrypt an email message** when given 90 minutes in which to do so. Furthermore, **one quarter of them accidentally exposed the secret they were meant to protect in the process, by sending it in email they thought they had encrypted but had not.**" [emphasis added]
- But hey, that was sixteen years ago, right? Surely we've perfected PGP/GPG since then? We'll see! :-)

Is PGP/GPG Something of A "Dead End?"

- In the spirit of full disclosure, some people will tell you that learning to use PGP may be a waste of your time. See for example:
 - Matthew Green's "What's the matter with PGP?"
<http://blog.cryptographyengineering.com/2014/08/whats-matter-with-pgp.html> (But also see the rejoinder at:
<https://pthree.org/2014/08/18/whats-the-matter-with-pgp/>)
 - Moxie Marlinspike's "GPG and Me"
<http://www.thoughtcrime.org/blog/gpg-and-me/>
 - "The World's Email Encryption Software Relies on One Guy, Who is Going Broke," <http://www.propublica.org/article/the-worlds-email-encryption-software-relies-on-one-guy-who-is-going-broke> [don't worry too much, he's gotten funding now]

If PGP/GPG Is "Dying," No One's Told Facebook That

 techcrunch.com/2015/06/01/facebook-now-supports-pgp-to-send-you-encrypted-emails/  

 [News](#) [Startups](#) [Mobile](#) [Gadgets](#) [Enterprise](#) [Social](#) [Europe](#) [Trending](#)  [Pinterest](#) [Instagram](#) [Microsoft](#) 

Facebook Now Supports PGP To Send You Encrypted Emails

Posted yesterday by [Frederic Lardinois](#) (@fredericl)

[Next Story](#)

CrunchBase

Edward Snowden	+
Email Encryption	+
Endgame	+
Facebook	+

NEWSLETTERS

- ✓ **TechCrunch Daily** Top headlines, delivered daily
- ✓ **TC Week-in-Review** Most popular stories, delivered Sundays
- ✓ **CrunchBase Daily** Latest startup fundings, delivered daily

Some Genuine Limitations of PGP/GPG

- PGP/GPG isn't perfect by any means. Real limitations include:
 - It can take some time and effort to learn to use PGP/GPG.
 - PGP/GPG isn't "built-in" to most messaging software; you need to manually add it.
 - Both you AND your correspondent need to use it if you want the protection of encryption
 - PGP/GPG doesn't encrypt message subject lines
 - It also doesn't protect you from traffic analysis attacks (an observer can still see who's sending a message, and who it's being sent to); that can be surprisingly revealing, in and of itself.
 - If you're careless, you can end up accidentally sending plain text when you meant to send cipher text.
 - If you forget your passphrase, you **will** lose everything encrypted with that passphrase. PGP/GPG is like a power tool or a firearm: if you fail to pay attention while using crypto/power tools/guns, you very well may get hurt.
Do NOT forget your passphrase!

On Balance, However...

- I still think it is worth while learning to use PGP/GPG, which is why I'm excited to have the chance to offer you this training today for M3AAWG attendees in Dublin.
- Having previously taught other people how to use PGP/GPG, I'm confident that I can teach you how to use it, too.
- I don't use PGP/GPG all the time, but I do use it when it makes sense to do so. I'd encourage you to adopt a similar policy.
- If it turns out you don't ever use it, that's fine too – I want the choice to be yours. Once you know how to use PGP/GPG, the choice **will** be yours.

Platforms

- People have been known to run PGP/GPG on pretty much everything or anything you can name.
- Today we're going to cover running PGP/GPG on:
 - Mac OS/X laptops (and Linux laptops, if we have any Linux users in the audience), and
 - MS Windows laptops.
- Once you have the outline from those platforms, you should be able to generalize that information to other operating systems (Linux, tablet operating systems, smart phones, etc.)

III. How PGP/GPG Works (In General)

PGP/GPG vs. Alternative End-To-End Crypto Options

- Before you sign up to become part of the PGP/GPG "club," you should know that there *are* alternatives for end-to-end email encryption, most notably S/MIME, and/or just use of a shared key.
- **S/MIME** relies on PKI personal certs, and has its own advantages and limitations. If you're interested in S/MIME, see the M3AAWG training I previously did on it; those slides are available at:

"Client Certs & S/MIME Signing and Encryption: An Introduction,"
<https://www.stsauver.com/~joe/maawg24/maawg24.pdf>

- You could also just encrypt a file using something like 7-Zip and a **symmetric key** (a key that both you and your correspondent know), and then send that encrypted file to your colleague.

Symmetrically Encrypting a File with 7-Zip*

- 7-Zip is an archiver that includes AES-256 symmetric encryption.
- Assume your secret message is in sample.txt, and your shared secret password is "SecretSecretFoo" (without quotes)
- Encrypt sample.txt with z7a creating sample.zip:
\$ 7za a -pSecretSecretFoo sample.zip sample.txt
- Recipient would then extract and decrypt sample.zip with:
\$ 7za e sample.zip
Enter password (will not be echoed) :
SecretSecretFoo

* 7-Zip is available free from <http://www.7-zip.org/download.html>

But There's The Rub....

- If you DON'T use something like PGP/GPG (or S/MIME), you and your correspondent both need to agree to use the **same secret key** (such as "SecretSecretFoo" in the preceding example). Remember, ncryption that uses the same key for both encryption and decryption is called "symmetric encryption."
- If you and your correspondent meet face-to-face, you can agree on a secret key that you'll use for all your future correspondence. But what if you've NEVER met a particular person face-to-face, yet you'd still like to be able to confidentially communicate with them? How do you bootstrap that initial "shared secret?"
- Or what if you want to communicate with **multiple people**: do you share the same common key with all of them? Ugh.
- Do you really want to manually set up personalized keys face-to-face with hundreds of people? I really doubt it.

Scaling Key Exchange: Symmetric Crypto Can't

Number of Correspondents	Symmetric crypto with a unique key for each pair of correspondents	Public key crypto, as used by PGP/GPG
2 (you and someone else)	1 key that you both know	2 key <i>pairs</i>
3 (you and two others)	$(3*2)/2=3$ keys, one for each pair	3 key <i>pairs</i>
12 people (you & 11 others)	$(12*11)/2=66$ keys	12 key <i>pairs</i>
100 people	$(100*99)/2=4,950$ keys	100 key <i>pairs</i>
500 people	$(500*499)/2=124,750$ key	500 key <i>pairs</i>

Understanding the math: consider the 100 person case. The first person needs a key for the other 99 people, ditto the 2nd person, the third person, etc. For a 100 people, that gets you to "(100*99)"

However, because we assume that the same key is used from sender A to receiver B, AND FROM sender B back to receiver A, that lets us divide the total number of unique keys required in half.

Why Not Just Share One Common Community-Wide Key?

- If you tried to short circuit this "itsy-bitsy" scaling problem by just having one common secret key shared by all correspondents, then all it would take would be for **one** untrustworthy correspondent to disclose the common key, and then your **entire** crypto network would be insecure!
- And in a network of 500 users, you're pretty safe in assuming that there's *at least* one person who's less than totally trustworthy (or at least somewhat careless).
- **Symmetric crypto just isn't the right answer.**
- **Symmetric crypto just doesn't scale, and it doesn't have a good way to bootstrap initial secret keys for remote colleagues, either.**

Asymmetric Cryptography

- PGP/GPG works by leveraging asymmetric ("public key") cryptography: instead of using the same key to encrypt AND decrypt, you'll use a **pair** of related keys, one to encrypt, and a corresponding (but different) one to decrypt.
- Half of your key pair is a **private key** that you'll never reveal to anyone. The other half is a corresponding **public key** that you can (and should!) share with the world.
- Knowing one of those keys (e.g., the public part) does NOT allow you to derive the other corresponding one.
- **Every user of PGP/GPG will normally have their own key pair.**

How Does Someone Get A PGP/GPG Key Pair?

- You make yourself one – anyone with the PGP/GPG software can do so.
- Creating a key pair for each of you is one of the things we'll do as part of today's session.

How Does Someone Get Someone Else's Public Key?

- PGP/GPG public keys are often distributed via public database servers known as "key servers." That said, some people may not know about key servers, or may choose not to use them.
- Those people will often put their public key on a web page, or they may simply email you their public key on request (they're free to distribute their public key any way they might like).
Note that you **MUST** share your publicly key with others *some* way or another. Unless you have a compelling reason not to, I recommend using PGP/GPG key servers.
- Why do some people not just use key servers? Three main reasons: they worry that spammers will harvest them to find email addresses to spam; they may find using key servers confusing; or they may be confused by the PGP/GPG "web of trust" concept.

Picking the RIGHT PGP/GPG Public Key

- **Anyone** can submit a PGP/GPG public key to a key server.
- Logical question, then: if **anyone** can submit a public key to a PGP/GPG public key server (and they can), "what keeps people from submitting **bogus** PGP/GPG public keys?"
- The answer to that is simple: **nothing**. Nothing prevents people from **submitting** totally bogus keys to a public key server.
- YOU are responsible for **selecting** the right/real one, assuming **any** of the available PGP/GPG keys are real (they might all be bad)
- But how to choose the right PGP/GPG key?
- There are essentially two approaches that people tend to use: you can rely on information received directly from the person who owns the key (e.g., a key fingerprint printed on a business card you received directly from that person), OR you can rely on the PGP/GPG "web of trust"

The PGP/GPG Web of Trust

- **Basic idea behind the web of trust: the owner of a PGP/GPG keypair asks others to digitally sign his/her public key.**
- Key signers sign after confirming the identity of a keyholder by inspecting a government-issued ID document (driver's license, passport, etc.). After signing a key, the signed key gets sent back to its owner.
- How do we know that those signing the key are who THEY claim to be? Well, ***their*** keys are in turn signed by other people... And the keys of those other people are in turn signed by other people...
- This creates an interwoven "web of trust" that ultimately acts to tie an identity to an associated key pair.
- The more broadly a key has been signed by other trusted signers, the more confidence you should have that the signed key belongs to the person whom it claims to represent.

NOTE: Signing a Key \neq "Vouching" For Someone

- From time to time I've run into people who confuse the signing of a key with "vouching" for someone's "trustworthiness" or "integrity." That's NOT what signing a PGP/GPG key is about.
- Signing a PGP/GPG key **IS** about asserting that there is a connection between:
 - a PGP/GPG key pair, and
 - an asserted identity.
- The actual person associated with an identity could be "as good as Mother Theresa" or "as evil as Hitler," you're NOT vouching for them just by signing their key! You ARE saying that you have personally made the connection between their real life identity and their keys.
- Then again, ***because of the confusion around this point***, I'd probably suggest NOT signing the keys of obviously evil people.

Is Signing PGP/GPG Keys "Required?"

- Signing PGP/GPG keys is NOT required. There is no PGP/GPG police who will "pull you over" for having an unsigned key.
- Some people NEVER bother to get their keys signed; when that's true, it is up to you to figure out an alternative way to ensure you've got the "right" key for that person.
- Likewise, you are NOT required to sign anyone else's key, although if you're a nice person, it's a kind thing to do for someone. We'll show you how to do this later in this training.
- In the other direction, no one else is required to sign YOUR key. If they do agree to sign it, they're doing you a favor, so be polite when asking and be gracious if someone declines to do so.

So Why Doesn't Everyone Use PGP/GPG?

- When we think about why PGP/GPG hasn't been more broadly adopted, you now are aware of several reasons:
 - Some people think they have nothing that needs protection (although they're probably wrong)
 - You need to download and install PGP/GPG, it doesn't come "built in" or "preinstalled"
 - It's relatively complex (not as hard as taking a class in differential equations, but not as simple as grilling a burger)
 - Your correspondents may not be using PGP/GPG
 - Key distribution may feel *ad hoc* or insecure (even though it can be perfectly secure when properly utilized), and the web of trust may seem somehow "kludgy" or *ad hoc*.
 - There are alternatives (such as S/MIME, or use of symmetric crypto) that may end up getting used instead of PGP/GPG.

Nonetheless, Let's Give It A Try Today

- The process isn't all that hard -- at least once you've gotten things setup and you've collected keys for all the people you routinely correspond with who are ALSO doing PGP/GPG
- Let's start by getting you setup...
- **These are tasks that you'll (hopefully) only need to do once.**

III. One-Time Setup Tasks

Versions of PGP/GPG

- There are several different versions of PGP/GPG.
- PGP ("Pretty Good Privacy") has now largely been supplanted by GNU Privacy Guard (GPG).
- Where possibly, we're going to illustrate use of the classic version of GNU Privacy Guard, Version 1.4.19 (as released on February 27th, 2015)
- I prefer classic GPG (e.g., GPG 1.x, NOT GPG 2.x) because GPG 1.x does **NOT** use pinentry. At one point I thought that I was the only fossil who felt this way, but in talking with GPG using colleagues, I've learned that I'm not alone. :-)
- If you want to install GPG 2.x, you can install it right alongside GPG 1.x, but we're not going to cover that today (except for folks who basically have no other option), such as our Windows-using colleagues.

Installation for the Various Platforms

- We are going to show installing GPG on both Macs [and Linux] and on Windows PC.
- **We'll begin with the Mac [and any Linux People]**
- Windows people, hang tight for a minute. After we get the Mac and Linux users get rolling, we'll come back to you.
- **On the Mac [and Linux], we're going to compile and install GNU Privacy Guard from source.**

MAC USERS

(And Linux Users, If Any...
You'll Start a Few Slides In)

Mac-1. Mac Users:

GNU Privacy Guard Installation

Installing GPG From Source for the Mac

- Normally, for ease of use, I would recommend simply installing the precompiled binary version of a program (any program!), rather than compiling from source.
- However, in this case, there are no precompiled binaries available for the classic version of GNU Privacy Guard for the Mac.
- GNU Privacy Guard is also security-sensitive software, so I'm happier building it from scratch, anyhow.
- Therefore, we have a little work to do, first...
- Hopefully, you all saw a note from me, forwarded by Amy, urging you to preinstall Xcode and the Xcode command line tools.
- **If you're a Mac user, did anyone NOT get that note, or NOT get that "homework" done?**

I'm Going to Assume We DO Have There's At Least One Mac User Who Didn't Get That Prep Work Done

- Therefore, let's go over this...
- To build GNU Privacy Guard on the Mac, you'll need to:
 - Be able to get to the Mac OS X terminal application on your Mac
 - Know the admin password for your Mac
 - Have Apple Xcode installed
 - Have the Apple Xcode command line tools installed
- We'll now take care of those points.

The Mac Terminal App

- We'll eventually move to a graphical point-and-click environment, but we're going to start by working at the command line.
- On the Mac, that means using the Mac Terminal app, which emulates an old DEC VT100-family terminal.
- If you don't routinely use the Mac Terminal app, you can find it at:

Finder → Applications → Utilities → Terminal.app

- Once you launch it, you'll be looking at a plain terminal window with a Unix command line shell prompt (the dollar sign). **Do NOT type the dollar sign when it is shown in subsequent commands!**

Mac Administrative User

- Mac OS X allows users to have different roles: you can be a regular user, for example, or an administrator with enhanced privileges.
- When you first installed Mac OS X on your laptop, one of the first things you did was to create an administrator account.
- For the purpose of installing GNU Privacy Guard, **you should be working from that administrator account. That means you should know the password for it.**
- Not sure if an account's an admin account? Go to Apple → System Preferences → Users & Groups and look to see if you're using an account tagged as "admin"

Install the Apple Developer Tools

- Compiling software on the Mac generally requires Xcode, the Apple developer tools. Anyone can register to become an Apple "developer" and you can then freely download Xcode.
- **IF** you didn't already install Xcode, you can **download Xcode 6.3.2 for free from Apple**. See the link located at:

<https://developer.apple.com/xcode/downloads/>

- Note that this is a **LARGE** download and may take a substantial amount of time over a shared wireless network. **If you've not previously downloaded and installed Xcode, you may want to try JUST installing the XCode Command Line tools, instead.**



Xcode

The complete toolset for building great apps.

[What's New](#)[Xcode IDE](#)[Interface Builder](#)[Features](#)[Downloads](#)

Download Xcode for Free

Xcode 6.3.2

This release includes the Xcode IDE, LLVM compiler, Instruments, iOS Simulator, the latest OS X and iOS SDKs, WatchKit and the Swift 1.2 programming language.

Updated: May 18, 2015

Build: 6D2105

SDK: iOS 8.3, OS X v10.10

[View in the Mac App Store >](#)



That link will take you over to...

Xcode

By Apple

Essentials

Open the Mac App Store to buy and download apps.



[View in Mac App Store](#)

Free

Category: [Developer Tools](#)

Updated: May 18, 2015

Version: 6.3.2

Size: 2.57 GB

Description

Xcode provides everything developers need to create great applications for Mac, iPhone, and iPad. Xcode brings user interface design, coding, testing, and debugging all into a unified workflow. The Xcode IDE combined with the Cocoa and Cocoa Touch frameworks, and the Swift programming language make developing apps easier and more fun than

[Apple Web Site](#) ▶ [Xcode Support](#) ▶ [Application License Agreement](#) ▶

[...More](#)

What's New in Version 6.3.2

Xcode 6.3.2 improves stability and fixes an issue that could result in slow compile times for complex Swift projects.

Xcode 6.3 includes Swift 1.2 and SDKs for OS X 10.10 Yosemite and iOS 8.3

[...More](#)

Screenshots

That link will take you to the App Store...

Installing The Command Line Tools

- **IF YOU HAVE INSTALLED Xcode but HAVEN'T installed the Command Line Tools**, you also need to download the command line tools. Do so by (a) starting Xcode, then (b) going to the Xcode→Preferences→Downloads tab and clicking "Install" next to the Command Line Tools entry.
- **IF YOU HAVEN'T INSTALLED Xcode (and don't WANT to!)**, you can try downloading JUST the command line tools by entering **\$ xcode-select -install**
That MAY be all you actually need.
- **Regardless of the approach you use**, signify that you accept the Command Line Tool license by entering in a Terminal window:
\$ sudo xcodebuild -license
Hit space, space, space, etc., to scroll down through the text of the legal agreement. When you get down to the very bottom, enter:
agree

Linux Users, You Should Start Paying Attention Here

- If you're a Linux person and you want to build GNU Privacy Guard from scratch, start paying attention here, please!
- The command line process is basically the same for Mac users and for you...

Downloading GPG 1.4.19 Source Code

- Download a copy of the GPG 1.4.19 source code by going to a Terminal window, and then entering (at the dollar sign prompt):

```
$ curl -O ftp://ftp.gnupg.org/gcrypt/gnupg/gnupg-1.4.19.tar.bz2  
(that's curl space dash oh, not curl space dash zero):
```

- Next, verify the checksum of that file (and thus its integrity):

```
$ shasum gnupg-1.4.19.tar.bz2  
5503f7faa0a0e84450838706a67621546241ca50  
gnupg-1.4.19.tar.bz2
```

That checksum should match what's shown above (and what's shown online at <https://lists.gnupg.org/pipermail/gnupg-announce/2015q1/000363.html>)

Unpack The GNUPG Distribution

- Uncompress the archive you downloaded:

```
$ bunzip2 gnupg-1.4.19.tar.bz2
```

This will leave you with the uncompressed archive file
gnupg-1.4.19.tar

- Untar that tar archive:

```
$ tar xfv gnupg-1.4.19.tar
```

That will unpack the GNU Privacy Guard distribution into the
subdirectory gnupg-1.4.19

Installing GPG

- Change directory down into the gnupg-1.4.19 subdirectory:
\$ cd gnupg-1.4.19
- Configure the package:
\$./configure ← **NOTE THE DOT at the start of that command!**
- Build and check the program:
\$ make
\$ make check
- Install gpg if all checks are reported as passing okay:
\$ sudo make install

Note: sudo will ask for your administrator password to run.

Confirm That GPG Is Installed

- **\$ gpg --version**

gpg (GnuPG) 1.4.19

Copyright (C) 2015 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later <<http://gnu.org/licenses/gpl.html>>

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg

Supported algorithms:

Pubkey: RSA, RSA-E, RSA-S, ELG-E, DSA

Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
CAMELLIA128, CAMELLIA192, CAMELLIA256

Hash: MD5, SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224

Compression: Uncompressed, ZIP, ZLIB, BZIP2

See A Short Summary of GPG Commands

- `$ gpg --help | more`

[...]

Syntax: `gpg [options] [files]`

Sign, check, encrypt or decrypt

Default operation depends on the input data

Commands:

<code>-s, --sign [file]</code>	make a signature
<code>--clearsign [file]</code>	make a clear text signature
<code>-b, --detach-sign</code>	make a detached signature
<code>-e, --encrypt</code>	encrypt data
<code>[etc]</code>	

See Also The GPG Unix Manual Page

- **\$ man gpg**

GPG(1)

GNU Privacy Guard 1.4

GPG(1)

NAME

gpg - OpenPGP encryption and signing tool

SYNOPSIS

gpg [--homedir dir] [--options file] [options] command [args]

DESCRIPTION

gpg is the OpenPGP only version of the GNU Privacy Guard (GnuPG). It is a tool to provide digital encryption and signing services using the OpenPGP standard. gpg features complete key management and all bells and whistles you can expect from a decent OpenPGP implementation.

This is the standalone version of gpg. For desktop use you should consider using gpg2 from the GnuPG-2 package

[etc]

Mac-2. Mac [and Linux] Users:

**Creating and Managing
YOUR OWN Keypair**

Create A Keypair For Yourself

- `$ gpg --gen-key`
[...]

Please select what kind of key you want:

(1) RSA and RSA (default)

(2) DSA and Elgamal

(3) DSA (sign only)

(4) RSA (sign only)

Your selection? **1**

RSA keys may be between 1024 and 4096 bits long.

What keysize do you want? (2048) **4096**

Requested keysize is 4096 bits

Key Duration/Validity Period

- Please specify how long the key should be valid.
[...]
Key is valid for? (0) **5y**
Key expires at Sat May 16 16:53:55 2020 PDT
Is this correct? (y/N) **y**
- **Notes:** You can pick whatever duration you like. I recommend five years as a reasonable key validity period. Some people like a shorter period (such as just 1 year), other people might do 20 years. I think 5 years is a reasonable compromise value.
- Make a mistake? Hit a control-C to abort and then re-run
\$ gpg --gen-key

Put In Your Name, Email, and An Optional Comment

- You need a user ID to identify your key [etc]:

Real name: **Joe St Sauver**

Email address: **joe@stsauver.com**

Comment: **Code 40**

You selected this USER-ID:

"Joe St Sauver (Code 40) <joe@stsauver.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? **o**

- **Note:** Enter your own information, obviously

Please also note that you do NOT need to enter a comment (this would be a good time not to just blindly copy what I enter)

Enter A Strong Passphrase

- You need a Passphrase to protect your secret key.

[enter a long/strong passphrase]

- **Notes:** Do NOT forget/lose this password. If you do, you'll be totally "out of luck." There is no magic backdoor for recovering your passphrase!

Also, while you DO want a long/strong passphrase, don't "go overboard" because you're going to be entering this all the time. If you make it TOO long/TOO strong, you'll regret it. You're going to be typing this password a LOT.

15-25 characters should be fine

Do Some Other Stuff While Your Keys Gets Created

- We need to generate a lot of random bytes. [etc]

.....+++++

.....+++++

gpg: key 36AD91D7 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb

gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model

gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u

gpg: next trustdb check due at 2020-05-16

pub 4096R/36AD91D7 2015-05-18 [expires: 2020-05-16]

Key fingerprint = 54A7 02D4 E156 1037 4ADF 2290 9D54 F6B4

36AD 91D7

uid Joe St Sauver (Code 40) <joe@stsauver.com>

sub 4096R/964600F3 2015-05-18 [expires: 2020-05-16]

Backup Your PRIVATE Key

- You routinely backup *everything*, right? Be sure to ALSO backup your SECRET (private) key
- To get a copy of your private key, enter:

```
$ gpg --export-secret-key --armor "joe@stsauver.com" > sec.key
```

Obviously, use YOUR email address, not mine, in this command.

- Save "sec.key" somewhere safe (perhaps on a thumb drive you can put in your bank's safety deposit box), then securely delete sec.key (\$ **rm sec.key**). Note that you will still need your passphrase to use your private key, so be sure to save THAT some place safe, too (ideally separate from the thumb drive, e.g., in your *other* bank safety deposit box)

Push Your PUBLIC Key to the Keyserver

- We'll now push your public key to the keyserver. The first step to doing this is finding the key ID of your key. List your keys with the `gpg --list-keys` command:

```
$ gpg --list-keys "joe@stsauver.com"
```

Notes: Substitute YOUR email address for my email address in this command. Look for the key ID (8 digit hexadecimal number) identifying **your** key; **my** key ID is 36AD91D7

- Now push your public key to one or more key servers (most of them periodically synchronize with each other). In my case I'd say:

```
$ gpg --keyserver pool.sks-keyservers.net --send-keys 36AD91D7
```

Sharing Your PUBLIC Key 1:1 With Someone

- Sometimes people either can't (or won't) work with public key servers. You can share your public key with them directly.

- For example, to export a copy of my public key to send to someone, I'd say:

```
$ gpg --export --armor joe@stsauver.com > temp.pubkey
```

```
$ cat temp.pubkey
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
Version: GnuPG v1
```

```
mQINBFVae+sBEAct1jsKMhRITthHI6qou4Dnr/O6frVPZkqBe+TG9oZddtiMAOZk  
NG8tKKB4Qn3BOOnxaN4OikAql96uFs9sfr9B0hBcMVuiOQPvO7c59o+W3kTIKMnN  
[etc]
```

- Cut and paste the entire public key block (including the dashed line bits) into a mail message to your potential correspondent.

A Quick Cut and Paste Review

- If you're working with GPG at the command line, you may be "cutting and pasting" stuff a lot.
- Most of you already know how, but just as a refresher:
 - Highlight a block of text by clicking at the start of what you want to copy, and then dragging the mouse downwards (while holding down the mouse button)
 - Copy the text by hitting Command-C (or by going to the menus and navigating to Edit → Copy)
 - Click where you want the text to go, then hit Command-V (or go to Edit → Paste)

Mac-3. Mac [and Linux] Users:

Getting and Managing Other People's Public Keys

Find a Correspondent's Public Key (And Automatically Downloading A Copy Of It)

- Assume we want to find the PGP public key for a correspondent, perhaps info@us-cert.gov

We'd search for that key by saying:

```
$ gpg --keyserver pool.sks-keyservers.net \  
--search-keys "<info@us-cert.gov>"
```

Notes: Put less than/greater than brackets around the email address as shown, within the enclosing double quote marks!

If there is more than one matching key, as in this case, you'll be asked to pick which one you want. Once you've selected the one you want, it will be downloaded and imported to your keyring

Getting A Correspondent's Public Key Shared 1:1, And Then Incorporating It Into Your Keychain

- Some correspondents may prefer to share their public key directly, rather than via a public keyserver. Perhaps they'll send it to you in an email message, or they may publish it on the web as the REN-ISAC does (<http://www.ren-isac.net/0x4DFD37BE.asc>)

To add such a key to your local key ring, put a copy of it into a file, perhaps called tempkey.txt. Any easy way to do that is to just copy and paste the key into that file using your favorite text editor (vi, emacs, Apple TextEdit.app, etc.)

- Then, at the \$ prompt, import that key by saying:

```
$ gpg --import < tempkey.txt
```

Mac-4. Mac [and Linux] Users:

Signing A Message You've Made

Getting Ready to Cryptographically Sign A Message

- Cryptographically signing a message proves that you are taking responsibility for it, and that it hasn't been tampered with during delivery (assuming the signature validates OK on the other end).
- To do this, begin by creating a message using a text editor of your choice (vi, emacs, or the Apple TextEdit.app).
- Let's save that file as "sample.txt"

Now Sign That Message...

- `$ gpg --clearsign < sample.txt > sample.gpg`

You need a passphrase to unlock the secret key for
user: "Joe St Sauver (Code 40) <joe@stsauver.com>"
4096-bit RSA key, ID 36AD91D7, created 2015-05-18

- Important Note: a digitally SIGNED message is **NOT ENCRYPTED**
- Signing provides cryptographic proof of attribution and non-tampering, **it does NOT provide protection against eavesdropping.**
- You **MUST** encrypt the message if you want to protect it against potential eavesdropping!

Looking at the clearsign'd Message

- **\$ cat sample.gpg**

```
-----BEGIN PGP SIGNED MESSAGE-----
```

```
Hash: SHA1
```

```
This is a sample message,
```

```
Joe
```

```
-----BEGIN PGP SIGNATURE-----
```

```
Version: GnuPG v1
```

```
iQIcBAEBAgAGBQJVC6oAAoJEJ1U9rQ2rZHXG2UQAIO6h/r6mBK8uwVmQTOCblxI  
NyCnyi4+OKUvW9LN5TsuDPM/kW9XsHaPDUiDgH6RB2pyJabFqbRDov894Q1TZ5UB  
lerfUbpPJeF6DwwTcZy/f94rGXPz3CKxfWNDcr30u8AYNWmnjuR5qarMab6HyOQg  
[etc]
```

```
-----END PGP SIGNATURE-----
```

- Note that the message has two parts: the original message body, and the associated signature part, all in one file. This is only one of multiple signature formats (but we're not going into that today)

Sending The clearsign'd Message

- Highlight the PGP message (**including the signature and the dashed lines**) with your mouse, and copy it (Cmd-C)
- Go to your favorite email client and compose a new message
- Paste the message into the body of the message (Cmd-V)
- Send the message as you normally would

Mac-5. Mac [and Linux] Users:

**Validating A Signed Message
You've Received**

Validating A Signed Message You've Just Received

- Received a PGP/GPG signed message sent "inline?" Highlight the PGP part of the message (**including the dashed lines**) with your mouse, and then copy it (Cmd-C)
- Go to your favorite editor, create a new file, and paste the message into file (Cmd-V). Save the file as temp.txt
- Run that file through GPG:

```
$ gpg < temp.txt
```

- Receive a PGP/GPG signed message sent as an **attachment**? Save the attachment(s) as a file, then run that file through GPG as shown above.

What You'll See For a Sample Valid Signature

- **\$ gpg < sample.out**

This is a sample message,

Joe

gpg: Signature made Mon Jun 1 17:20:21 2015 PDT using RSA key ID 36AD91D7

gpg: Good signature from "Joe St Sauver (Code 40) <joe@stsauver.com>"

- You're looking for "Good signature"
- You're also looking to make sure the message came from who you think it should have come from

Mac-6. Mac [and Linux] Users:

**Encrypting a Message To
Protect Against Eavesdropping**

Getting Ready to Encrypt A Message

- Encrypting a message protects that message against eavesdropping
- To do this, begin by creating a message using a text editor of your choice (vi, emacs, or the Apple TextEdit.app).
- Let's save that file as "sample.txt"

Now Encrypt That Message...

- **\$ gpg --encrypt --armor < sample.txt > sample.gpg**
You did not specify a user ID. (you may use "-r")

Current recipients:

Enter the user ID. End with an empty line: **ren-isac@iu.edu**

gpg: 1AF9AF6A: There is no assurance this key belongs to the named user [...]

It is NOT certain that the key belongs to the person named in the user ID. If you **really** know what you are doing, you may answer the next question with yes.

Use this key anyway? (y/N) **y**

Current recipients:

2048g/1AF9AF6A 2005-08-08 "REN-ISAC <ren-isac@iu.edu>"

Enter the user ID. End with an empty line: **<just hit return>**

Side Bar: "What's 'ASCII armor'?"

- PGP/GPG can produce binary files, or 'ASCII armored' files, an easily transmitted ASCII-encoded file (think of ASCII armor as a "base64"-like encoding using printable ASCII characters only)
- If you ever go to cut and paste a PGP/GPG file and see something that looks like an ugly binary mess, you're probably looking at a file that was NOT sent in ASCII armor format.
- Simple rule for you: ALWAYS use PGP/GPG ASCII armor format when sending an encrypted file.

Some Notes About Encrypting

- When you are sending an encrypted message to someone, you need to **specify the recipient of that message** (e.g., you're encrypting the message with YOUR CORRESPONDENT's public key)
- If GNU Privacy Guard doesn't want to let you encrypt the message with their key (perhaps because it doesn't have a copy of it), do you actually have a copy of your correspondent's public key? If not, go get it, either from a key server or directly from him or her.
- If you think you DO already have their public key in your key ring, check to see if you're got a typo in their email address
- If your correspondent has a unique string in their name (perhaps they're the only person named "Freddy" in your PGP/GPG key chain), you can just enter that unique string to select their key.
- If you plan to keep a carbon copy of your encrypted message for reference, be sure to **ALSO** encrypt with **your own** key as an additional recipient.

Sending The Encrypted Message

- **\$ cat sample.gpg**

```
-----BEGIN PGP MESSAGE-----
```

```
Version: GnuPG v1
```

```
hQIOAzmmxFoa+a9qEAgArFjMBpCyt+bQcssD4Io2Jdc40tbpxCmgpzlBQ6i3Circ  
4yrfJ4IvI/6T0lX5Tadw5lSwy8mZinz1Nbuo/T56g5aVQ1PLHGIhcVcJiytCJAoj  
JQR/uNKnk43FNUY+DSwI/Ok3dl6/D9fIeBR+cL8gZG64dOfmPmXOpzhoNNpZHI9s  
/GHj+6QP9S6dht+CpKw+DL+qz04PiIk7L/dMdhlnfKAc+e1TGw9FnRdoIxUoYryH  
0fJDowVKNxdPd9j47nqDd9GvmDDJED2WhhFSa4p66E07n0o7hgIO2i93PIQJ9Nyb  
[etc]
```

```
-----END PGP MESSAGE-----
```

- Highlight the PGP message **(including the dashed lines)** with your mouse, and then copy it (Cmd-C)
- Go to your favorite email client and compose a new message
- Paste the message into the body of the message (Cmd-V)
- Send the message as you normally would

Quick Notes About Sending Encrypted Messages

- If you're sending an ASCII armored message, you do NOT need to send it as an attachment. It's plain text, generally safe to send in the body of a regular message.
- If you are sending sensitive content in an encrypted message, "perhaps" it would be a good idea to NOT tip your hand by using an informative Subject line for your message.

In general, **NO subject line** is the best Subject line for encrypted messages.

- Some sites may refuse ALL encrypted email messages since they normally can't scan those messages for malware. If you have a rejected message returned for that reason, contact your recipient via an alternative method.

Mac-7. Mac [and Linux] Users:

**Decrypting an Encrypted Message
You've Just Received**

Decrypting An Encrypted Message You've Just Received

- PGP encrypted message sent "inline?" Highlight the PGP part of the message (**including the dashed lines**) with your mouse, and then copy it (Cmd-C). Now Go to your favorite editor, create a new file, and paste the message into file (Cmd-V). Save the file as temp.txt
- Run the message through GPG:

```
$ gpg < temp.txt
```

You will need to enter your passphrase to decrypt the message

- PGP/GPG encrypted message sent as an **attachment**? Save the attachment as a file, then run it through GPG as shown above.

Notes About Decrypting An Encrypted Message

- Encrypted messages normally aren't (and can't be!) scanned for malware, phishing, spam, or other undesirable content.

You've been warned. Be careful! You may be potentially decrypting dangerous content when you decrypt a PGP/GPG message!

- NOTE: Now you understand one reason why I just show decrypting messages to the screen on the previous slide. While you could decrypt a message to a file, doing so increases the chance that you will end up decrypting a troublesome/unsafe file.
- **In general, strongly encourage your correspondents to only send you plain text files through PGP/GPG.**

Mac-8. Mac [and Linux] Users:
Signing Another User's Public Key

Signing Another User's Public Key

- Import the user's public key to your key ring if you haven't already done so. See Section Mac-3, earlier in this talk.
- Confirm that the key you have for the user is actually the key the user uses (check the fingerprint of the key). For example:
\$ gpg --list-keys johnsmith@example.com
- Confirm the user's identity by inspecting the user's government-issued identification (his passport, driver's license, etc.)
- If that all checks out okay, sign the key:
\$ gpg --sign-key johnsmith@example.com
[...]
Really sign? (y/N) **y**
You need a passphrase to unlock the secret key for [your userID]:
[enter your passphrase here]
- Export the key you've signed, and send that file to the user:
\$ gpg --export --armor johnsmith@example.com

Why Not Send The Key You've Signed Directly to the Keyserver?

- Generally speaking, you should NOT send a key you've signed directly to a public keyserver.
- Why?
- Some users may not want their public key distributed via the public keyserver infrastructure.
- Since you generally can't remove a key once it's been published to the public keyservers, please do NOT send another user's key to the public keyservers unless they explicitly ask you to do so.
- Send the signed public key back to the user for them to handle.

Mac-9. Mac [and Linux] Users:

**Importing Your Own Key
After It Has Been Signed**

Updating Your Key Ring With A Key That Someone Has Signed For You

- Once you've received a newly signed key, you will likely want to update your key ring with that new signature.
- Save the signed key you've received to a file, such as temp.key
- Import/merge that key:

```
$ gpg --import < temp.key
```

Updating Your Key As Saved On The Public Keyserver

- When you've received a newly signed key, you may also want to update your key on the public keyserver.
- After merging the signed key into your own keychain, update the key by pushing the signed key to the keyserver as described previously (see "Push Your PUBLIC Key to the Keyserver" in section Mac-2)

Mac-10. Mac [and Linux] Users:

PRACTICE!

Okay Mac/Linux Users, Now Practice For A Few Minutes!

- You've now got all the skills you need to do simple PGP/GPG messages. We're now going to work with the Windows folks a little
- Pair up with another Mac [or Linux] user, and do the following:
 - a) create a key pair for yourself if you haven't already done so
 - b) get a copy of your partner's public key
 - c) send a signed message to your partner
 - d) send an encrypted message to your partner
 - e) verify the signature of the message you received from your partner
 - f) decrypt the message you received from your partner
 - g) if so inclined, sign your partner's key and send it back to them

**WINDOWS
USERS**

Windows-1:

Windows GNU Privacy Guard Installation

Installing The Windows Gpg4win Binary

- Most attendees should have received a note last week asking them to download Gpg4win 2.2.4 (30Mbyte version) from

<http://www.gpg4win.org/download.html>

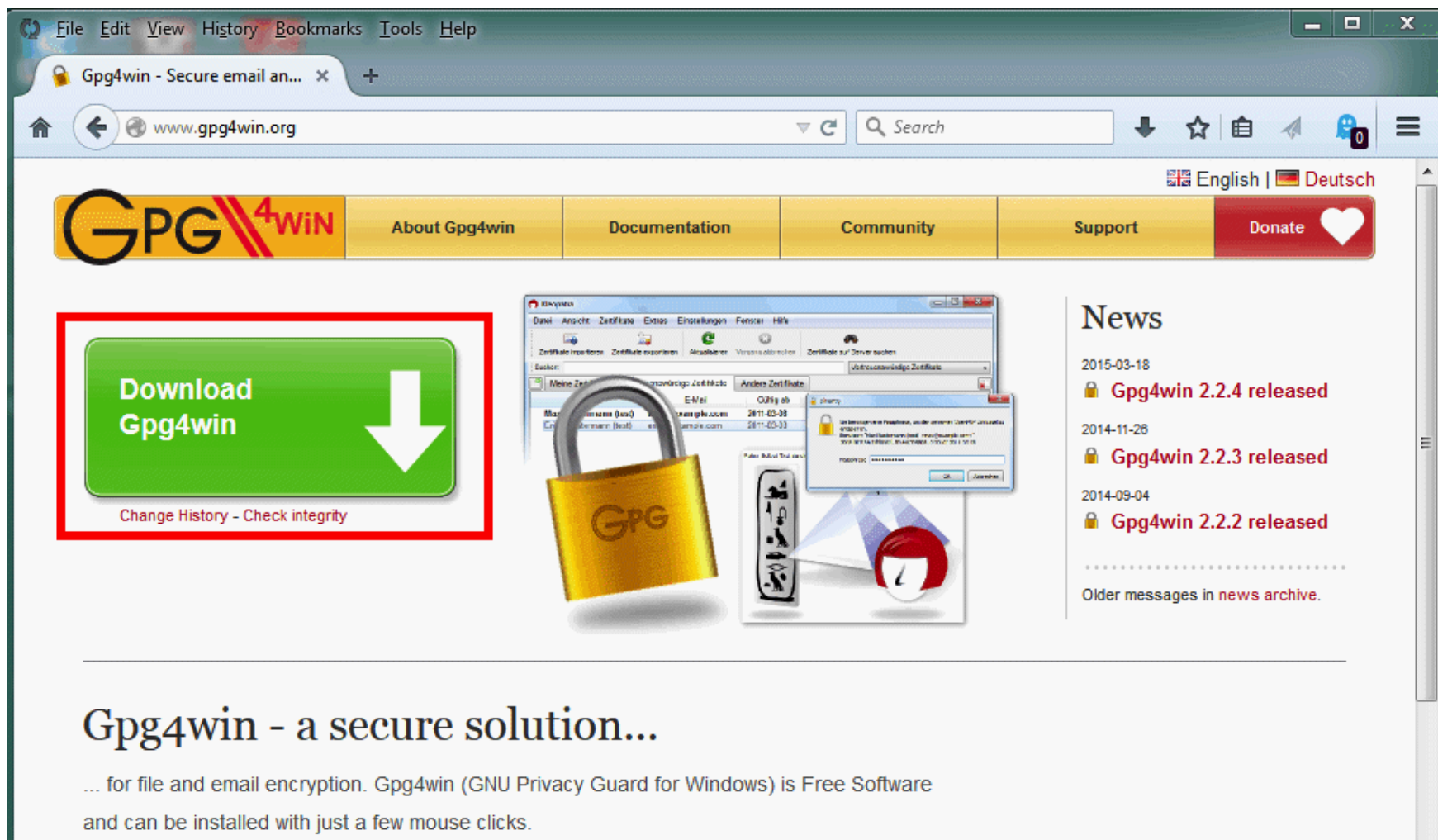
If you've not already done so, please do so now.

- **This is the official Windows implementation of GNU Privacy Guard.**

Why Do A Binary Version For Windows Users?

- I'd generally have preferred to do a built-from-source version for Windows users, just as we did for Mac users, but there are a variety of complexities associated with doing so that makes that less practical.
- In a somewhat ironic twist, for example, Gpg4win, the binary we'll be using, actually gets cross-compiled on a Linux system, not built "native" on Windows
- See <http://www.gpg4win.org/build-installer.html> for details – it just isn't realistic/practical to try to walk you through it for today's session.
- We're going with the binary as a result, sorry.

The Gpg4Win Top Level Web Site's Download Link



The screenshot shows the Gpg4Win website in a web browser. The browser's address bar displays "www.gpg4win.org". The website's header includes a navigation menu with links: "About Gpg4win", "Documentation", "Community", "Support", and a red "Donate" button with a heart icon. The main content area features a large green button with a white downward arrow and the text "Download Gpg4win". This button is highlighted with a red rectangular border. Below the button, the text "Change History - Check integrity" is visible. To the right of the button is a graphic showing a yellow padlock with "GPG" on it, and several overlapping windows from the Gpg4Win application. On the right side of the page, there is a "News" section with a list of releases: "2015-03-18 Gpg4win 2.2.4 released", "2014-11-26 Gpg4win 2.2.3 released", and "2014-09-04 Gpg4win 2.2.2 released". At the bottom of the page, the text "Gpg4win - a secure solution..." is followed by a description: "... for file and email encryption. Gpg4win (GNU Privacy Guard for Windows) is Free Software and can be installed with just a few mouse clicks."

Download Gpg4win

Change History - Check integrity

News

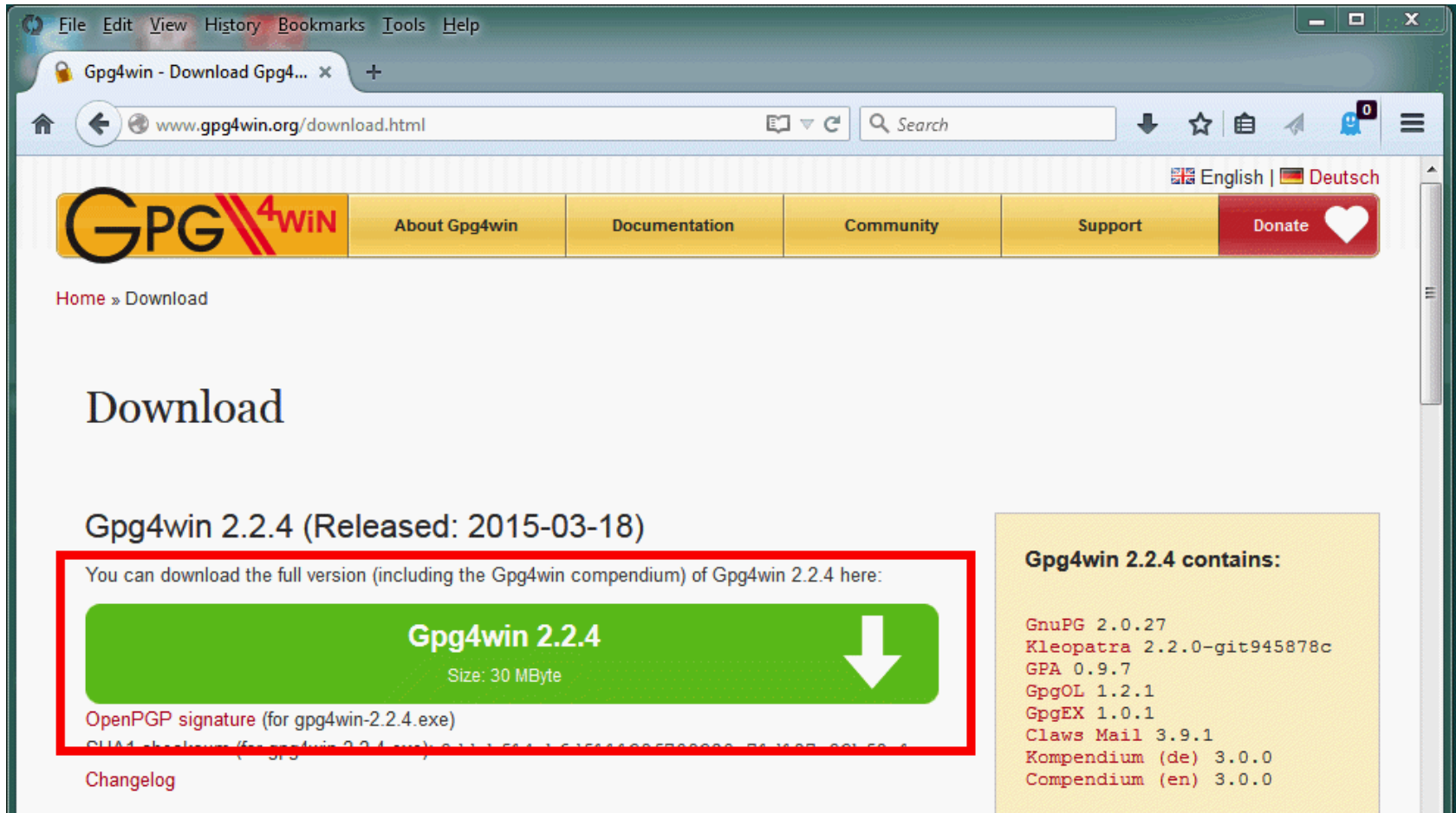
- 2015-03-18 Gpg4win 2.2.4 released
- 2014-11-26 Gpg4win 2.2.3 released
- 2014-09-04 Gpg4win 2.2.2 released

Older messages in [news archive](#).

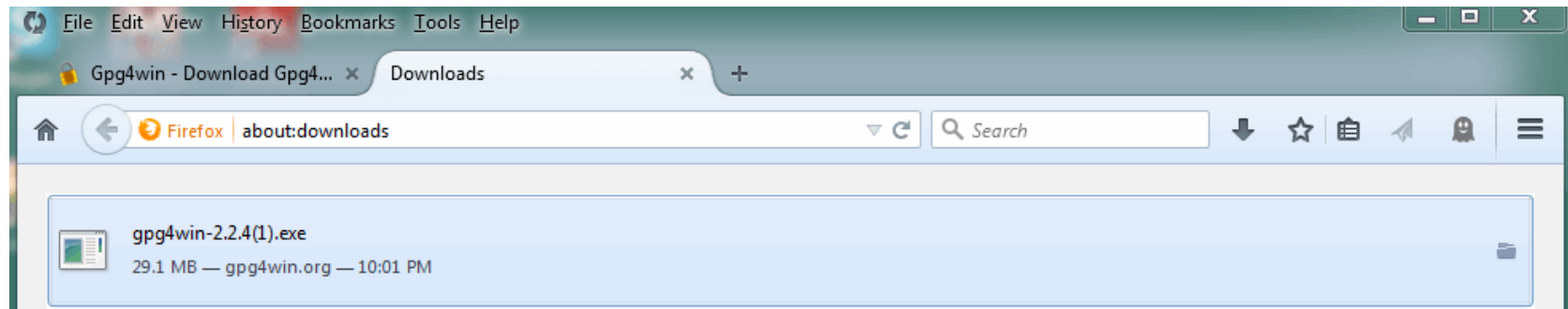
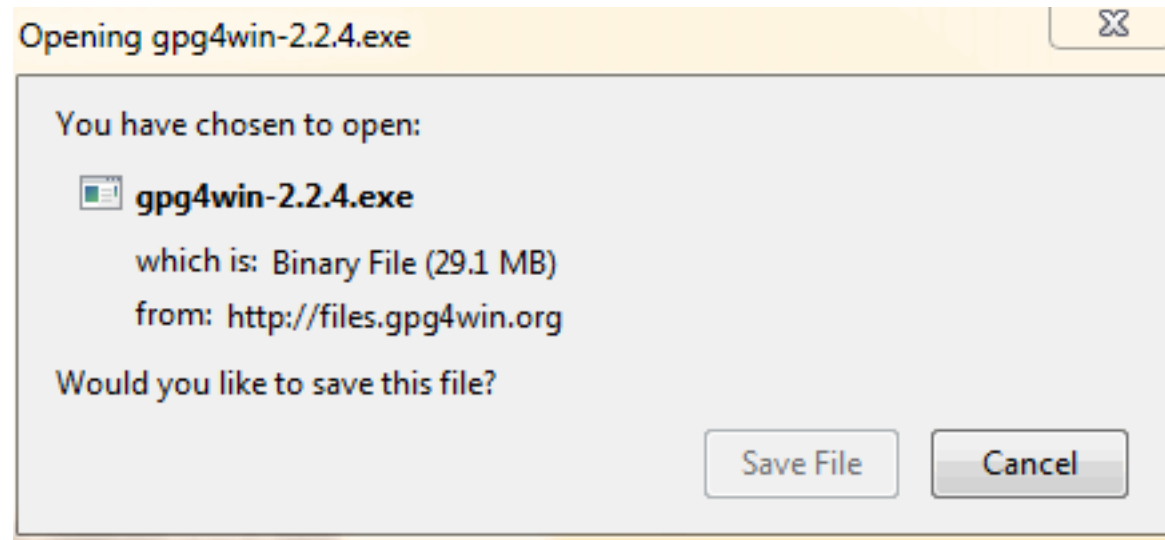
Gpg4win - a secure solution...

... for file and email encryption. Gpg4win (GNU Privacy Guard for Windows) is Free Software and can be installed with just a few mouse clicks.

There Are Several Download Options; Unless Space Is Really Tight, Do The Full One



Normal Firefox Download Process, Yielding...



Verifying The Distribution's Integrity

- To check that the binary has not been tampered with, you can check it's sha1 checksum value. For information on how to do this, see: <https://support.microsoft.com/en-us/kb/889768>

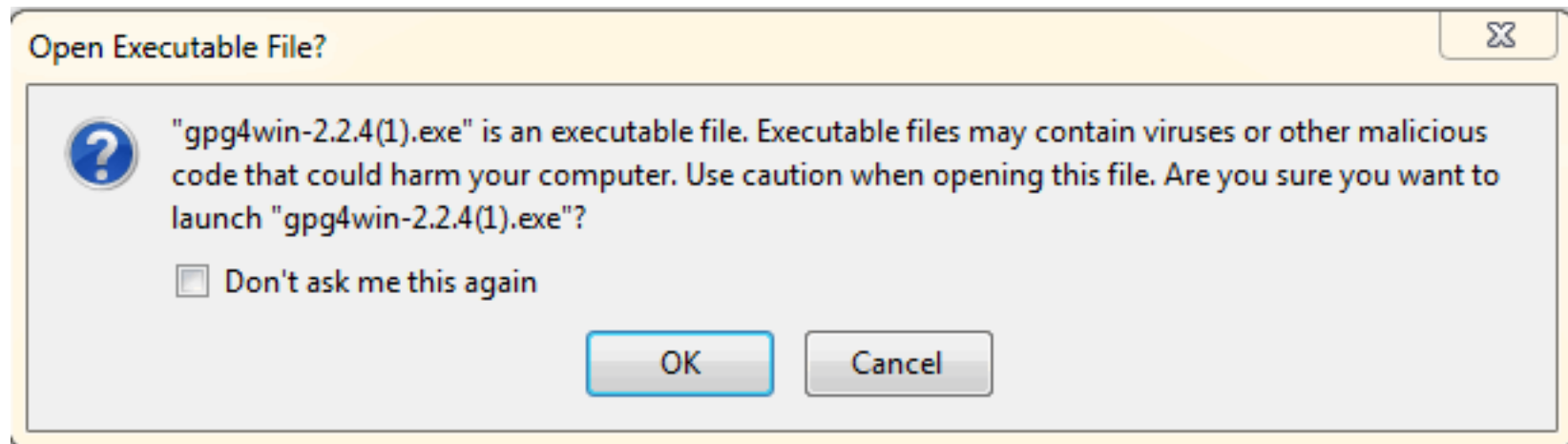
The sha1 checksum you see should be

8ddcbf14eb6df11139f709320a71d197a83bf9e1

(and as shown at <http://www.gpg4win.org/download.html>)

- However, note that the Gpg4Win installers are signed with a globally trusted cert, which may incline you to skip or deprioritize this step (See <http://www.gpg4win.org/package-integrity.html>)

Anyhow, When Ready to Proceed, Just Double Click The Downloaded File to Start the Installer



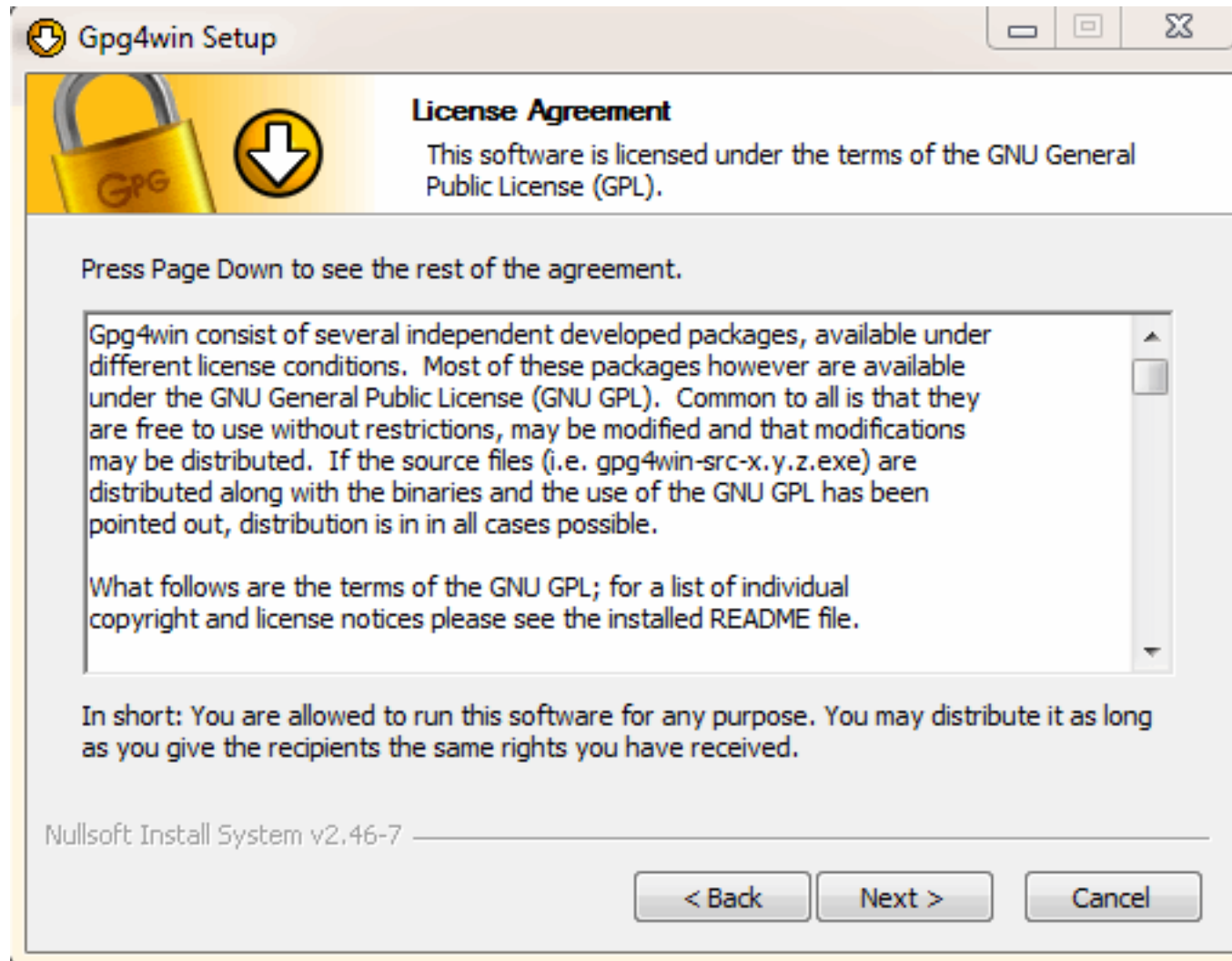
Note: Depending on the version of Windows you're running, you may need to explicitly allow the installer to modify your computer (please allow this, if asked).

You may also be asked to pick a preferred language.

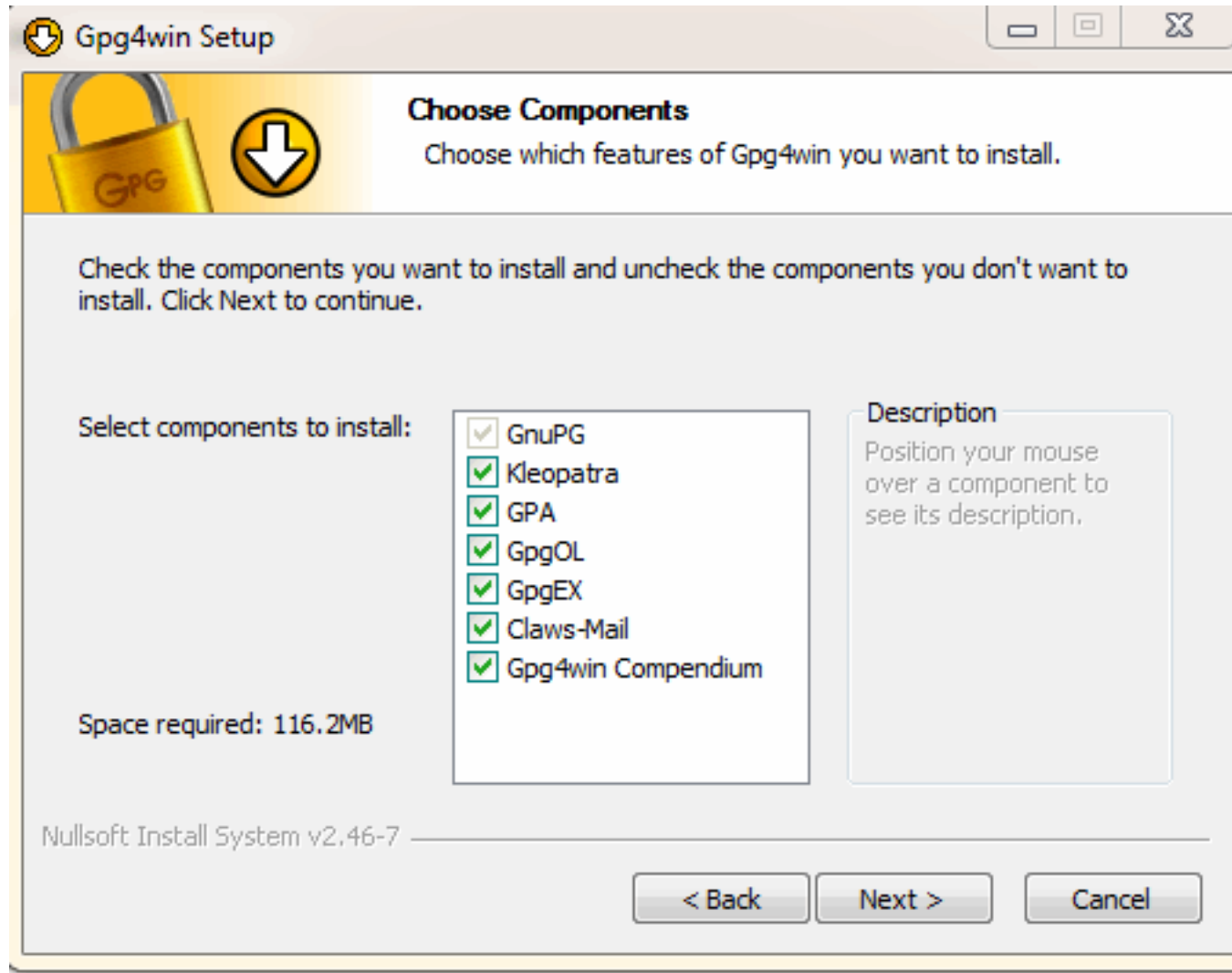
Opening Screen



Accept The GNU GPL

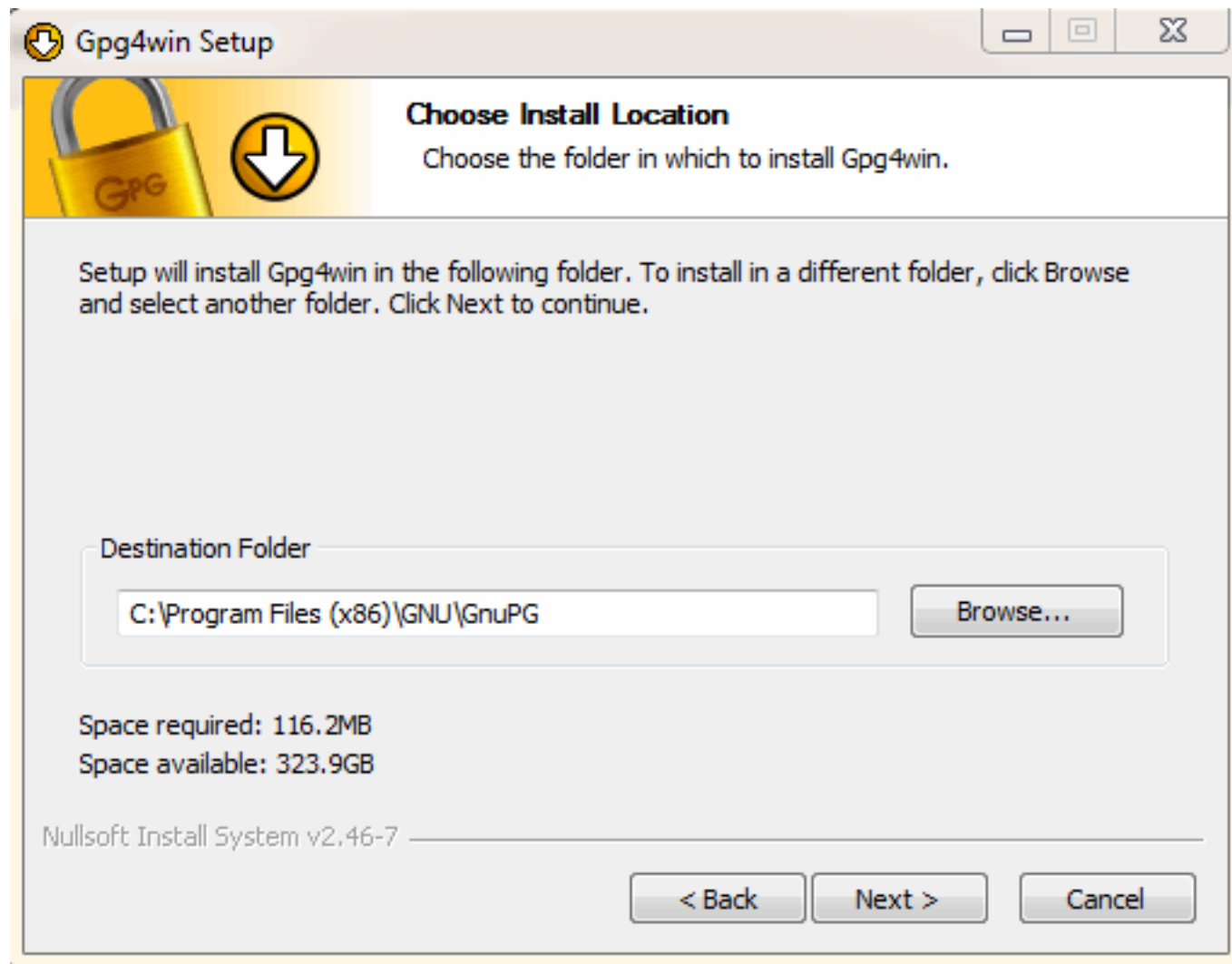


Choose Components To Install (Go Ahead and Choose Everything)

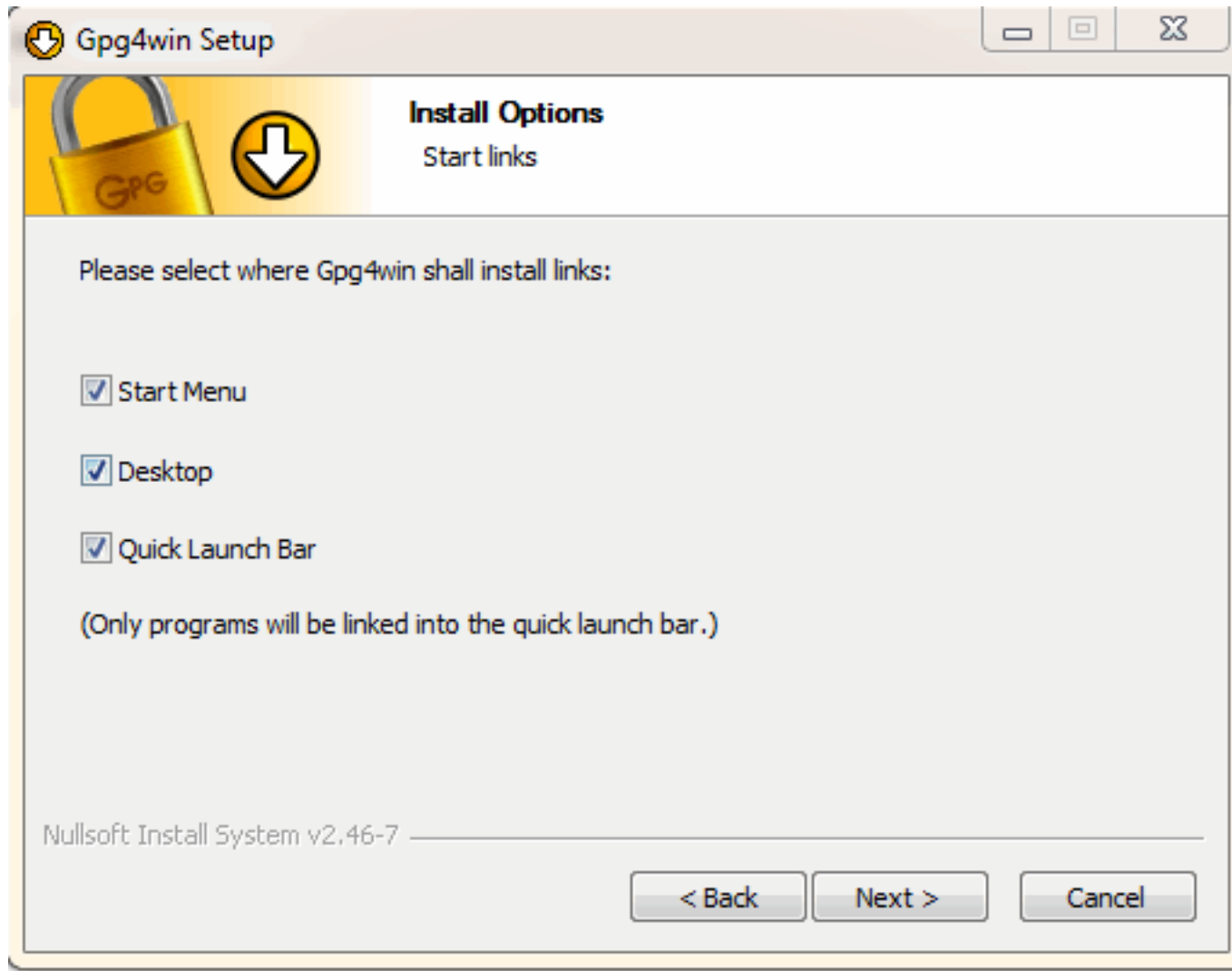


Pick An Install Location

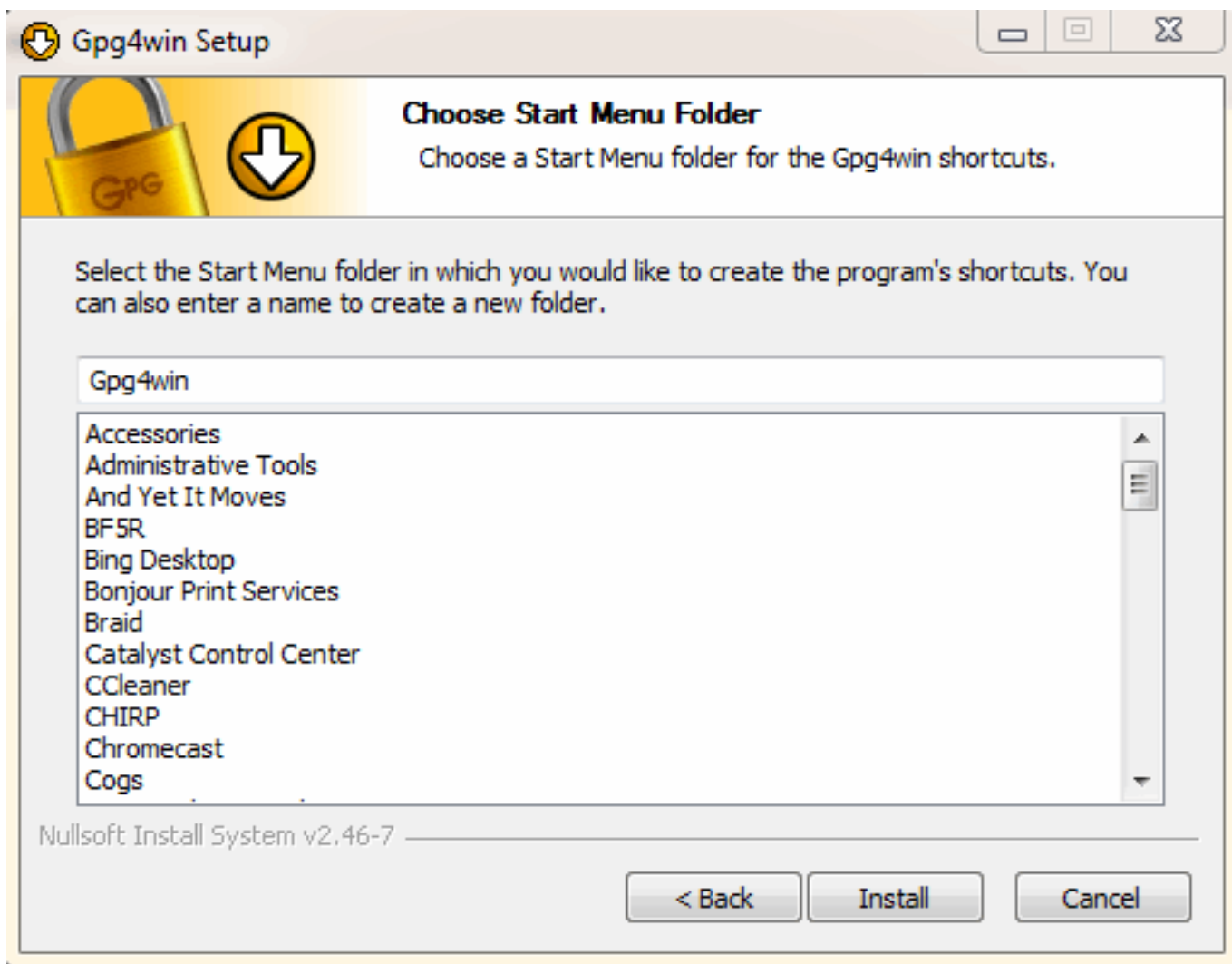
(I Suggest Accepting the Default Location)



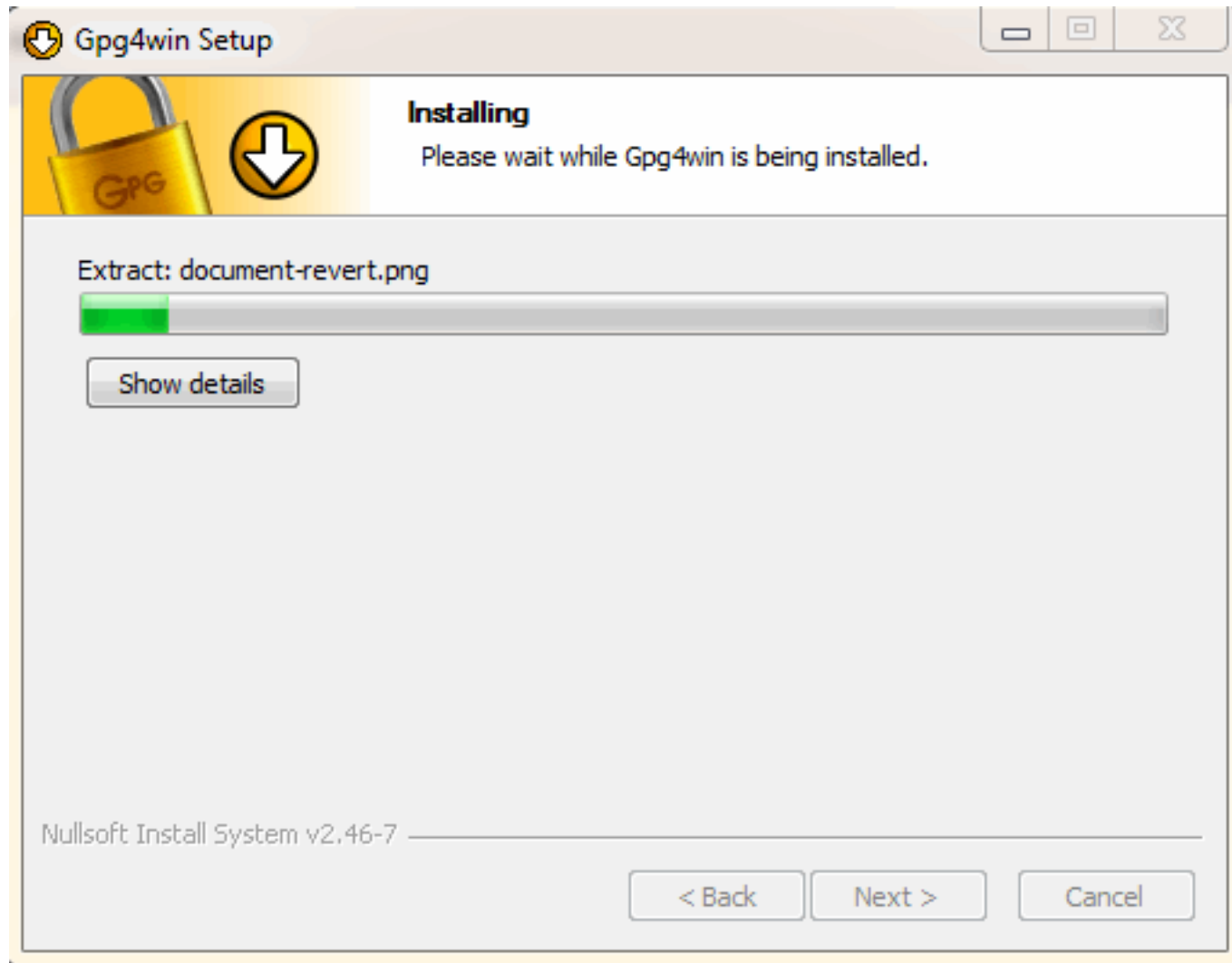
Stick Start Links "Everywhere"



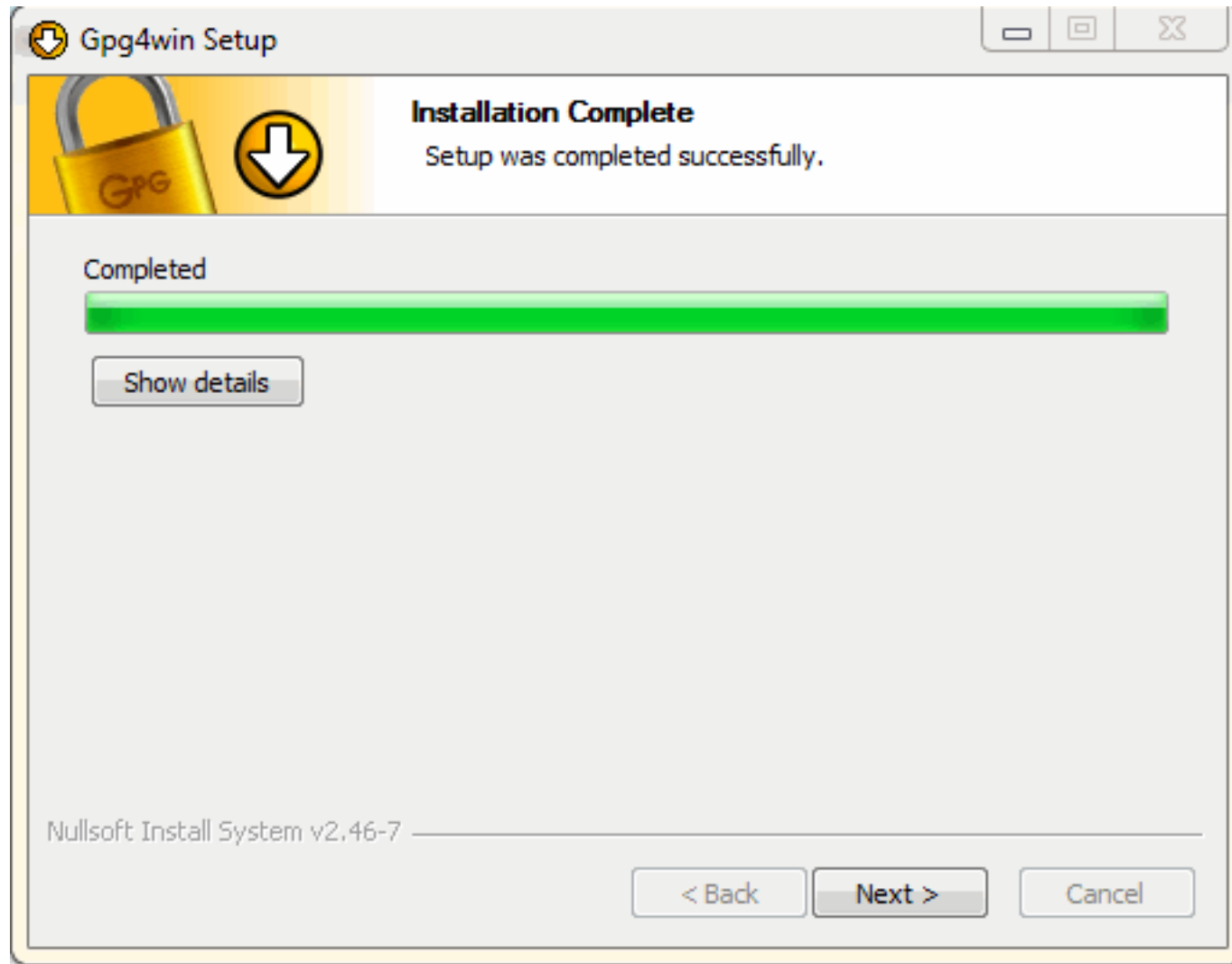
Confirm The Default Start Menu Folder Name



Installing...



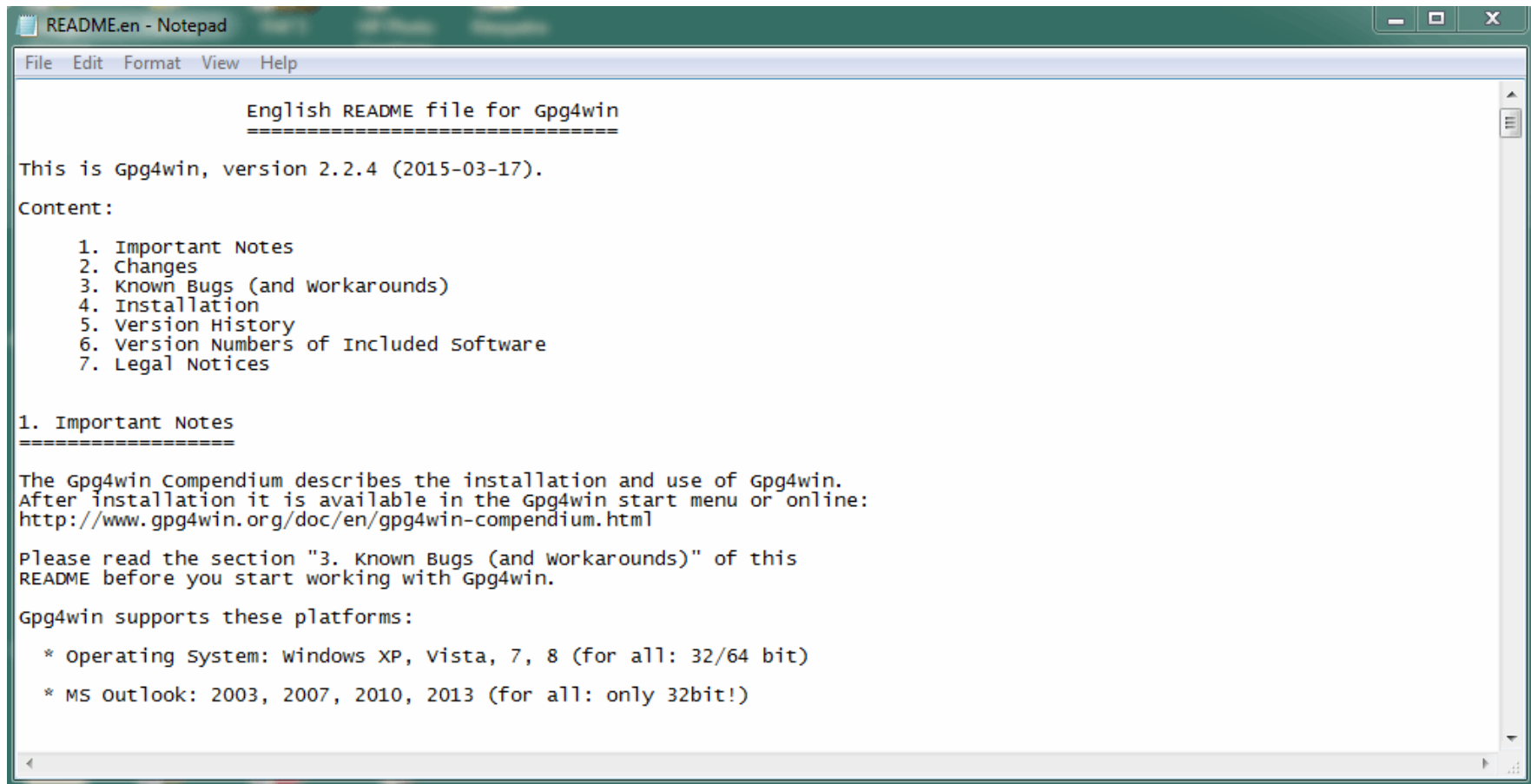
Normal End of Installation



Setup Process' Last Step Shows You the README



The README file



```
English README file for Gpg4win
=====

This is Gpg4win, version 2.2.4 (2015-03-17).
Content:
  1. Important Notes
  2. Changes
  3. Known Bugs (and workarounds)
  4. Installation
  5. Version History
  6. Version Numbers of Included Software
  7. Legal Notices

1. Important Notes
=====

The Gpg4win Compendium describes the installation and use of Gpg4win.
After installation it is available in the Gpg4win start menu or online:
http://www.gpg4win.org/doc/en/gpg4win-compendium.html

Please read the section "3. Known Bugs (and workarounds)" of this
README before you start working with Gpg4win.

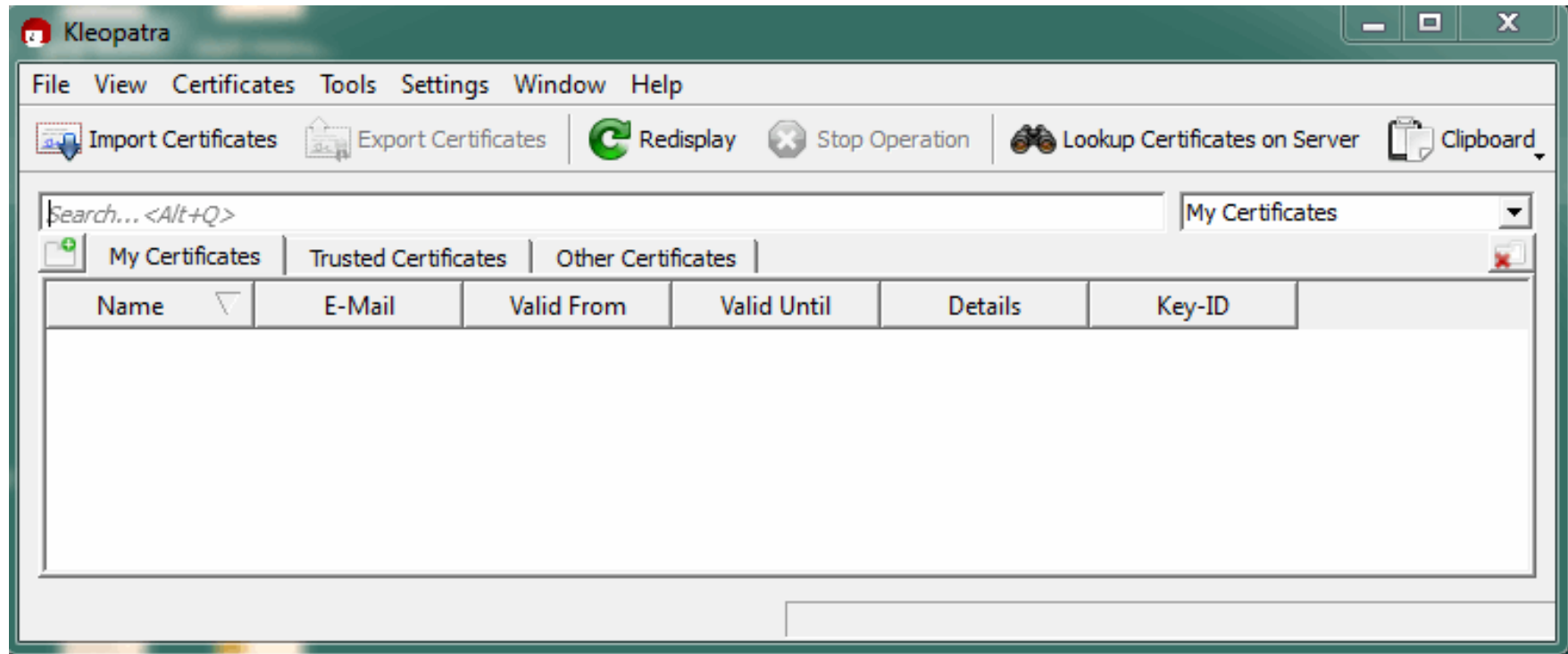
Gpg4win supports these platforms:
  * Operating System: Windows XP, Vista, 7, 8 (for all: 32/64 bit)
  * MS Outlook: 2003, 2007, 2010, 2013 (for all: only 32bit!)
```

Note: The "Gpg4win Compendium" == the Gpg4win user manual

Windows-2:

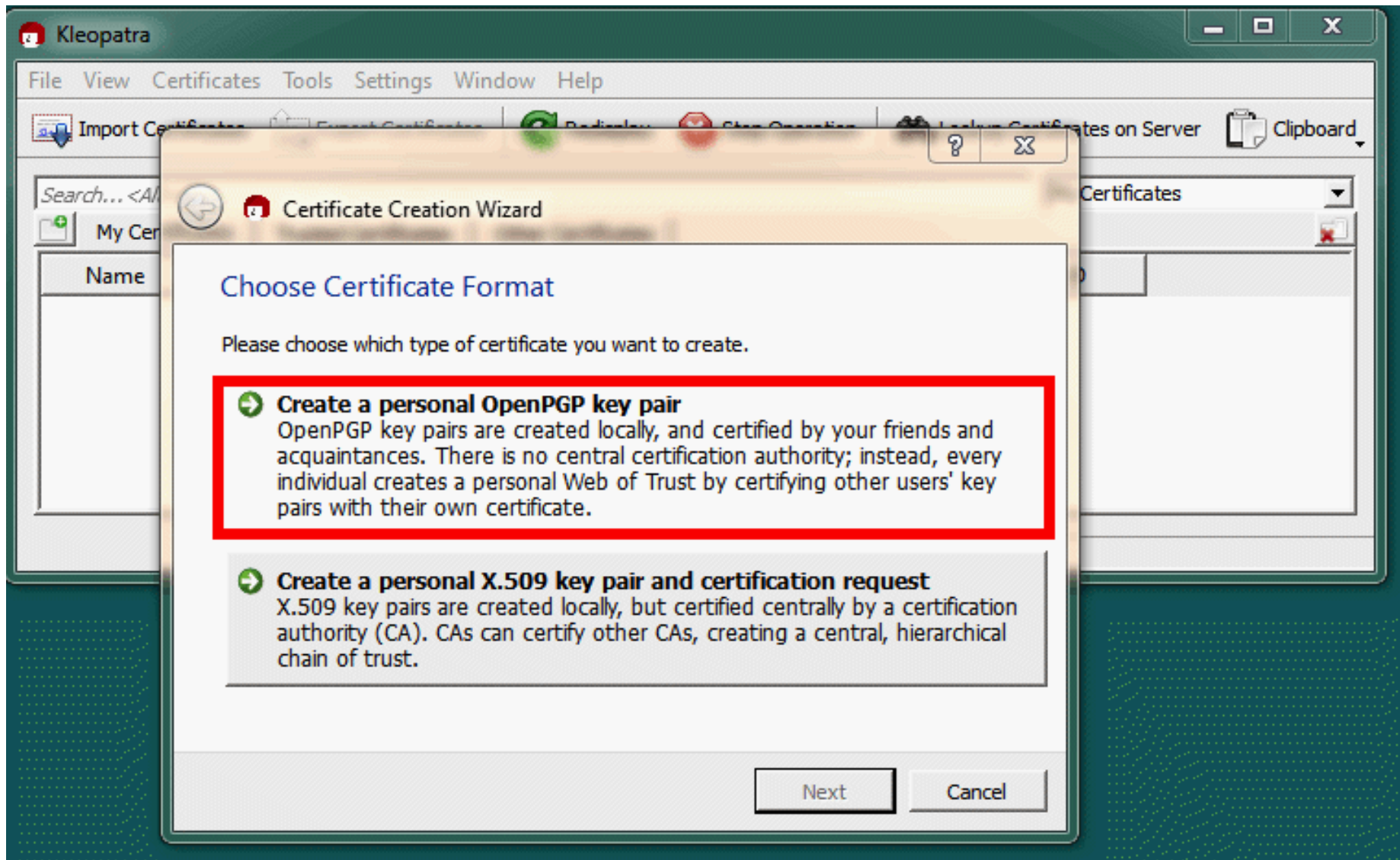
**Creating and Managing
YOUR OWN Keypair**

Start Kleopatra From Your Desktop or From The "All Programs" Menu

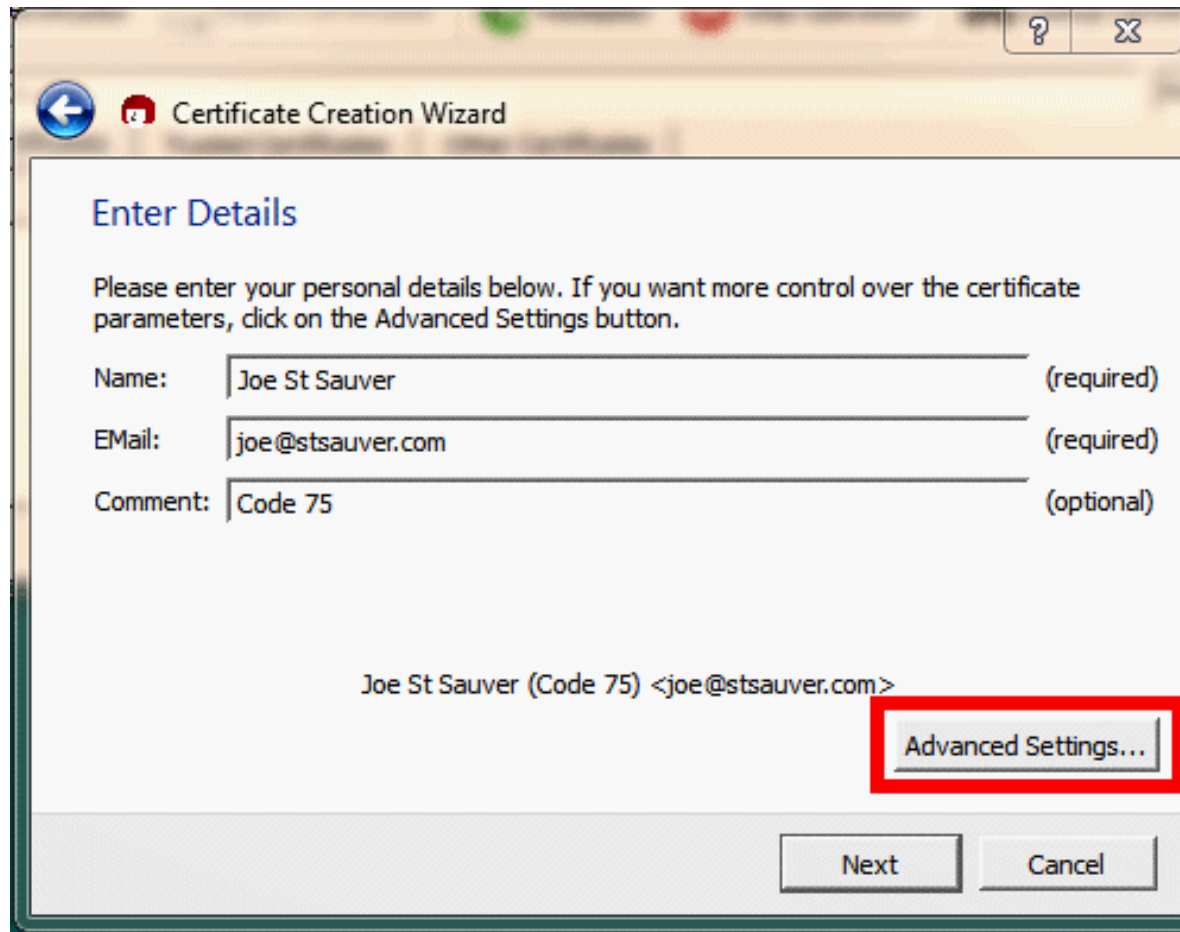


Note: You may be prompted to create a key pair the first time you run Kleopatra; if not, simply go to **File → New Certificate**

Choose Your Key's Format: Select OpenPGP



Create Your Keypair



The image shows a Windows-style dialog box titled "Certificate Creation Wizard". It has a title bar with a question mark and a close button. The main area is titled "Enter Details" and contains instructions: "Please enter your personal details below. If you want more control over the certificate parameters, click on the Advanced Settings button." There are three input fields: "Name:" with the value "Joe St Sauver" and "(required)" label; "EMail:" with the value "joe@stsauver.com" and "(required)" label; and "Comment:" with the value "Code 75" and "(optional)" label. Below these fields, the text "Joe St Sauver (Code 75) <joe@stsauver.com>" is displayed. A button labeled "Advanced Settings..." is highlighted with a red rectangle. At the bottom right, there are "Next" and "Cancel" buttons.

Enter Details

Please enter your personal details below. If you want more control over the certificate parameters, click on the Advanced Settings button.

Name: Joe St Sauver (required)

EMail: joe@stsauver.com (required)

Comment: Code 75 (optional)

Joe St Sauver (Code 75) <joe@stsauver.com>

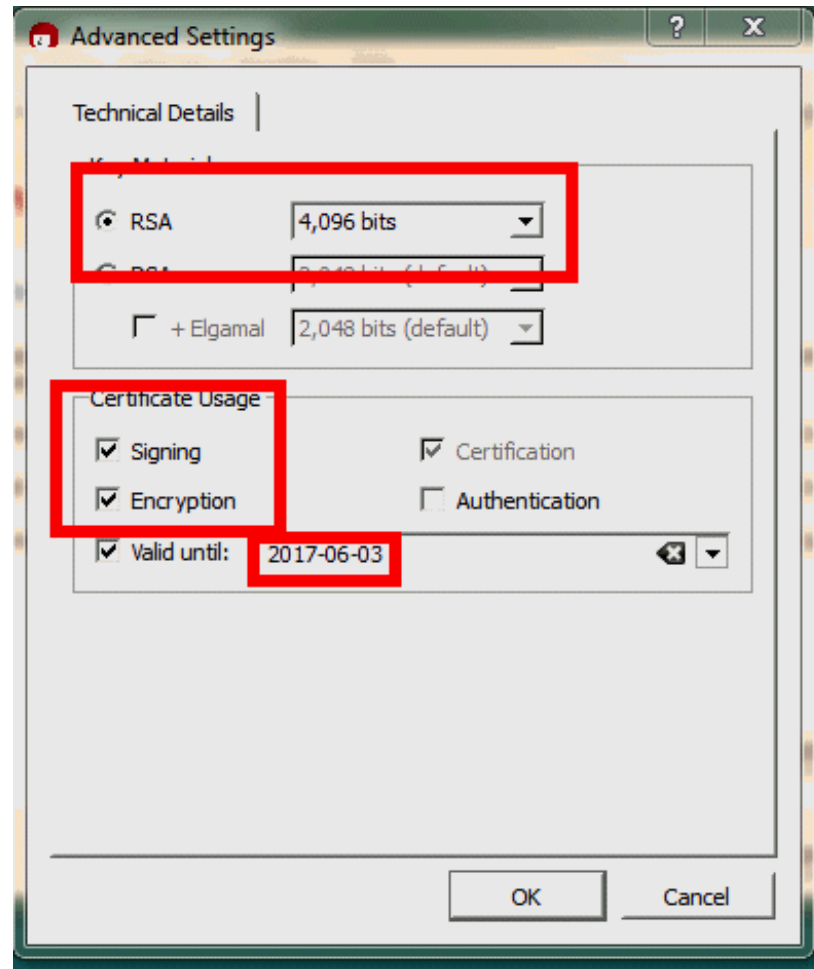
Advanced Settings...

Next Cancel

Note: Enter **your own** information; a comment is optional.

Note: Do NOT just click NEXT! Click on Advanced Settings, instead!

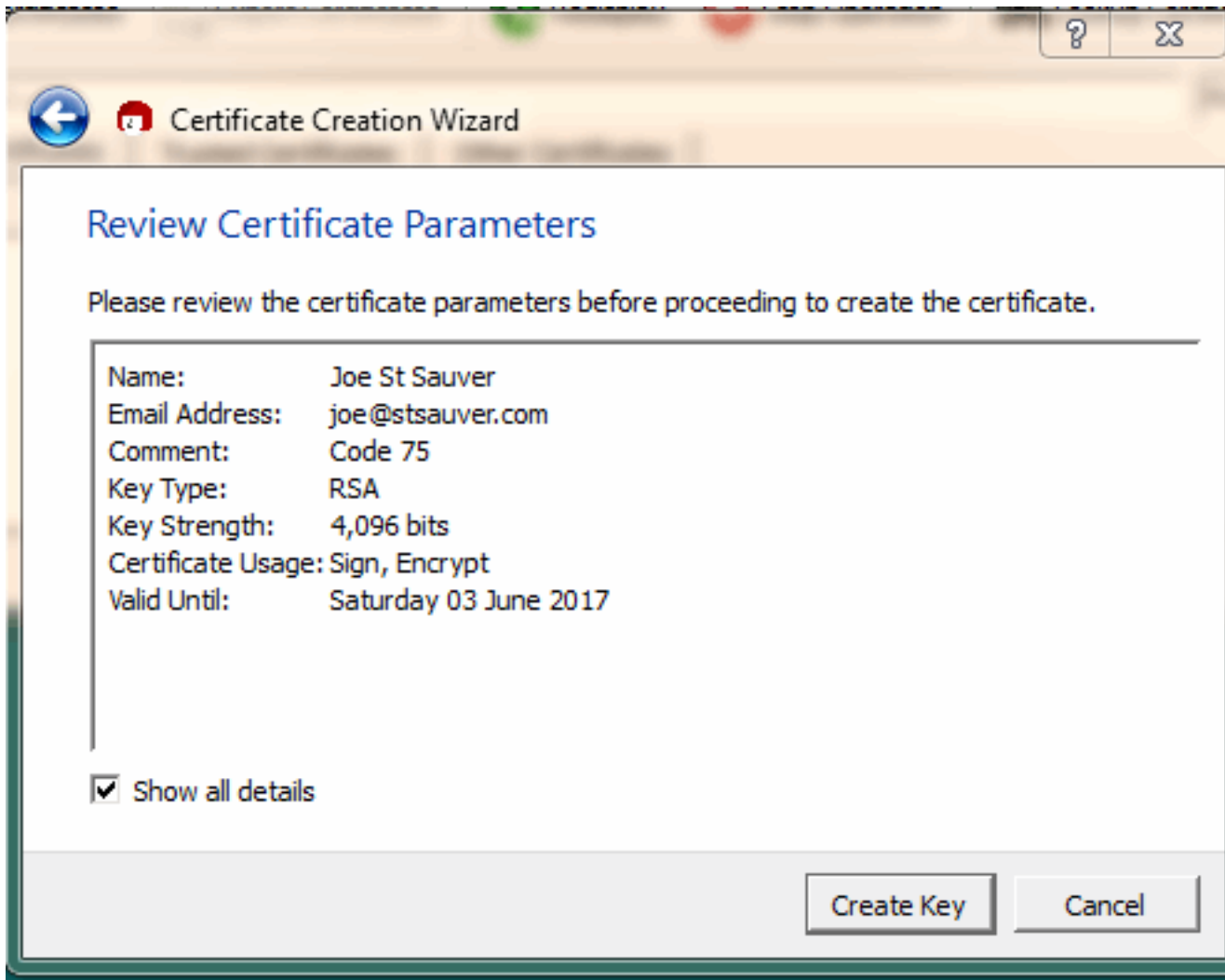
Create Your Keypair: Advanced Settings



Note: Do an RSA 4096 bit key, used for signing and encryption

Note: Pick a reasonable expiration date (1-5 years is usually fine)

Review Certificate Parameters



The screenshot shows a Windows-style dialog box titled "Certificate Creation Wizard" with a back arrow icon. The main heading is "Review Certificate Parameters". Below it, a message says "Please review the certificate parameters before proceeding to create the certificate." A list of parameters is displayed in a bordered box: Name (Joe St Sauver), Email Address (joe@stsauver.com), Comment (Code 75), Key Type (RSA), Key Strength (4,096 bits), Certificate Usage (Sign, Encrypt), and Valid Until (Saturday 03 June 2017). At the bottom left, there is a checked checkbox labeled "Show all details". At the bottom right, there are two buttons: "Create Key" and "Cancel".

Review Certificate Parameters

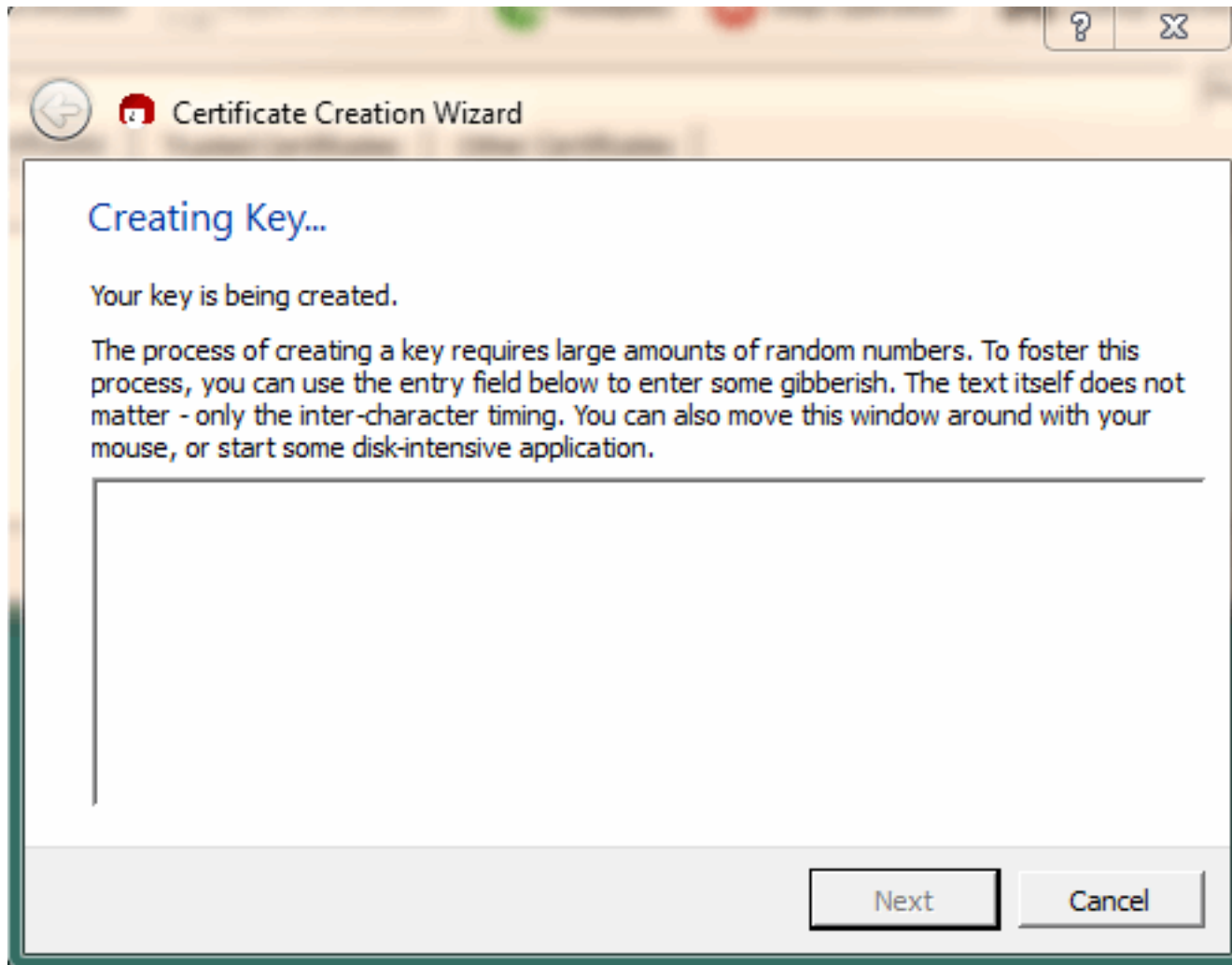
Please review the certificate parameters before proceeding to create the certificate.

Name:	Joe St Sauver
Email Address:	joe@stsauver.com
Comment:	Code 75
Key Type:	RSA
Key Strength:	4,096 bits
Certificate Usage:	Sign, Encrypt
Valid Until:	Saturday 03 June 2017

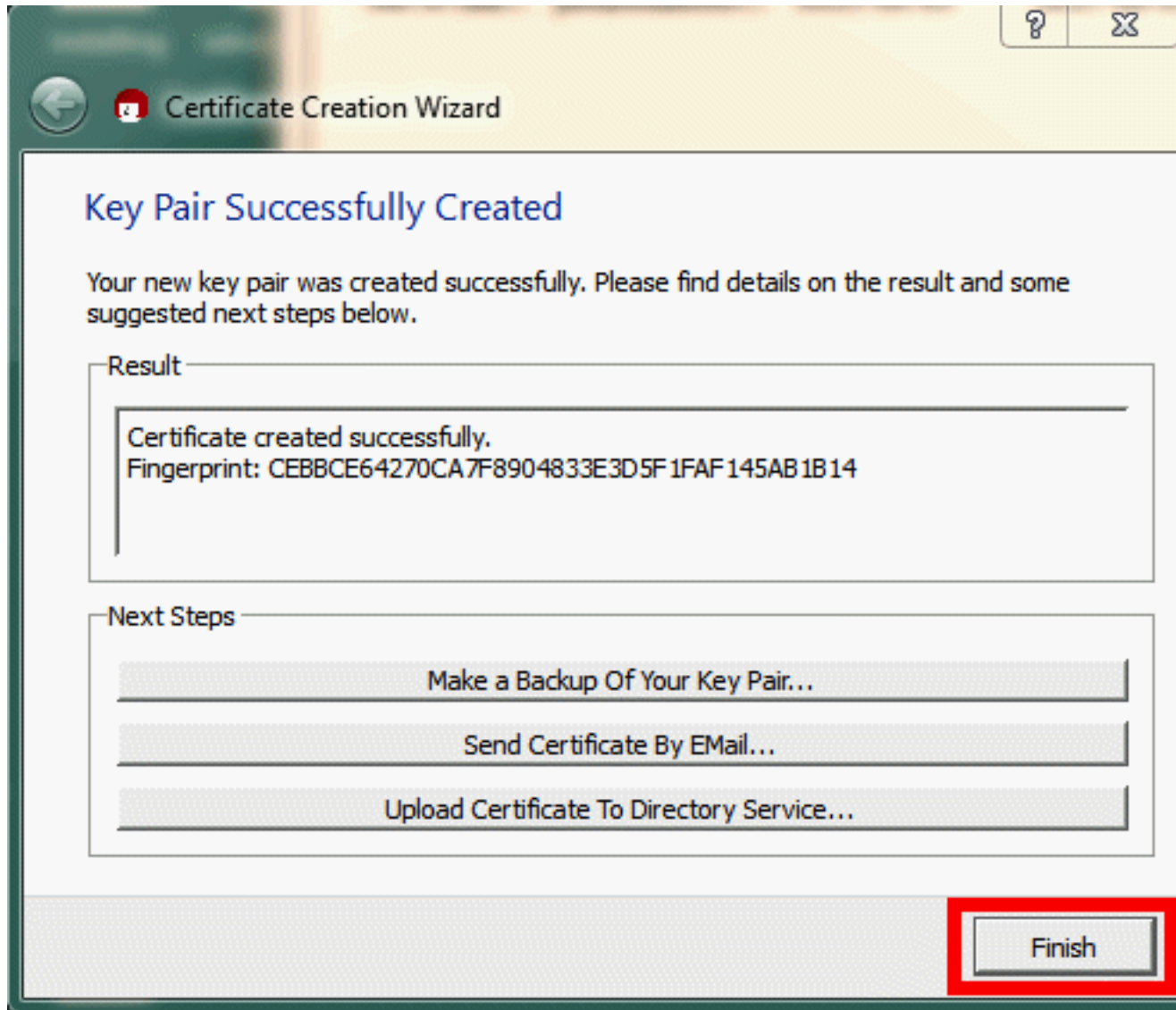
☒ Show all details

Create Key Cancel

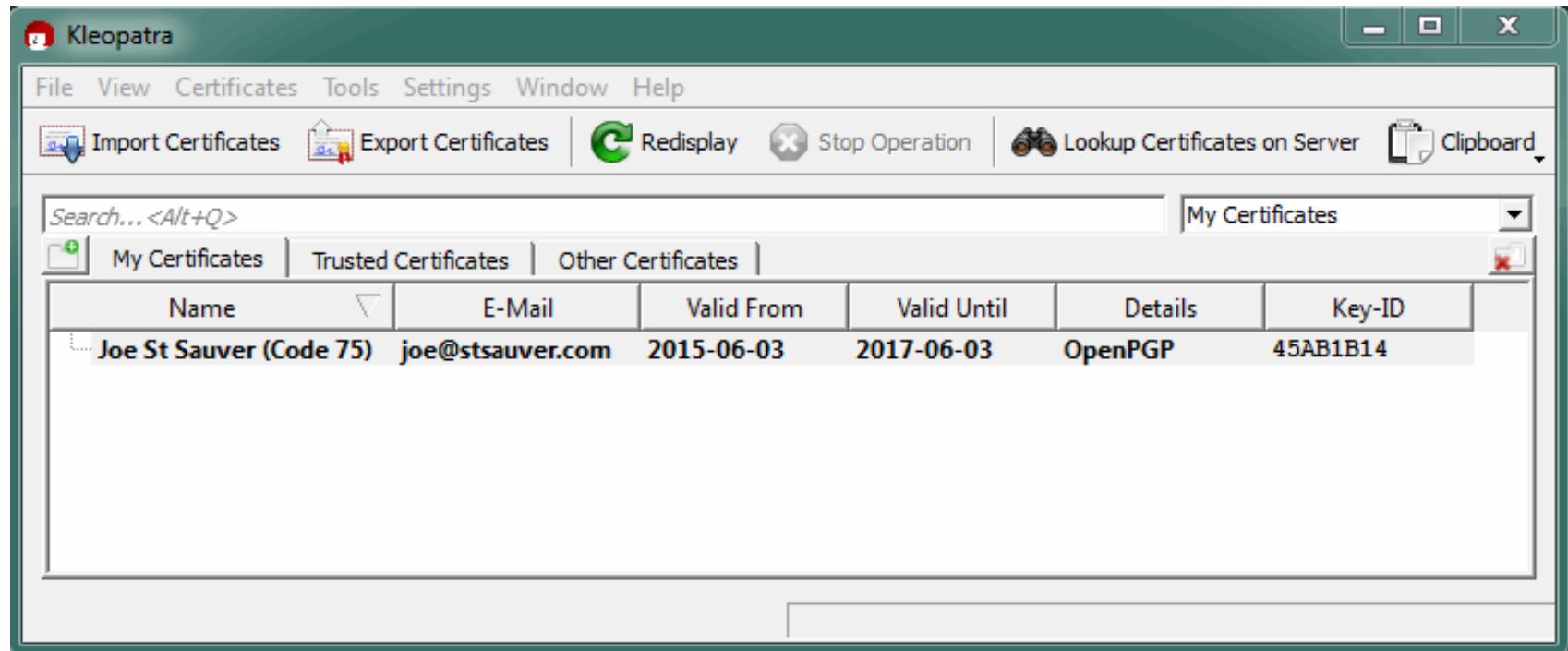
Do Stuff While Your Keys Are Being Created



Key Pair Successfully Created

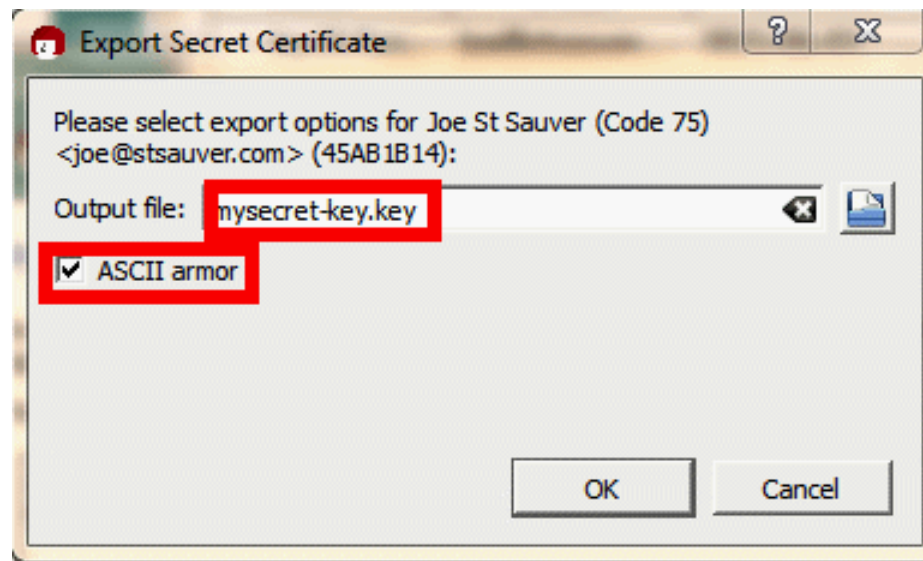


My New Key in the Kleopatra Certificate Display



Export Your SECRET Key For Backup Purposes

Select your new keypair in the Kleopatra Certificate Display Window, then go to File → Export Secret Key



Provide a file name, and confirm the directory where you'll be saving this file. Also be sure to check the "ASCII armor" box. Save this file somewhere safe, such as on a thumb drive in your safe deposit box. Safely save your passphrase somewhere else.

Now Push Your PUBLIC Key To the Keyservers

Select your new keypair in the Kleopatra Window, then go to File → Export Certificates to Server



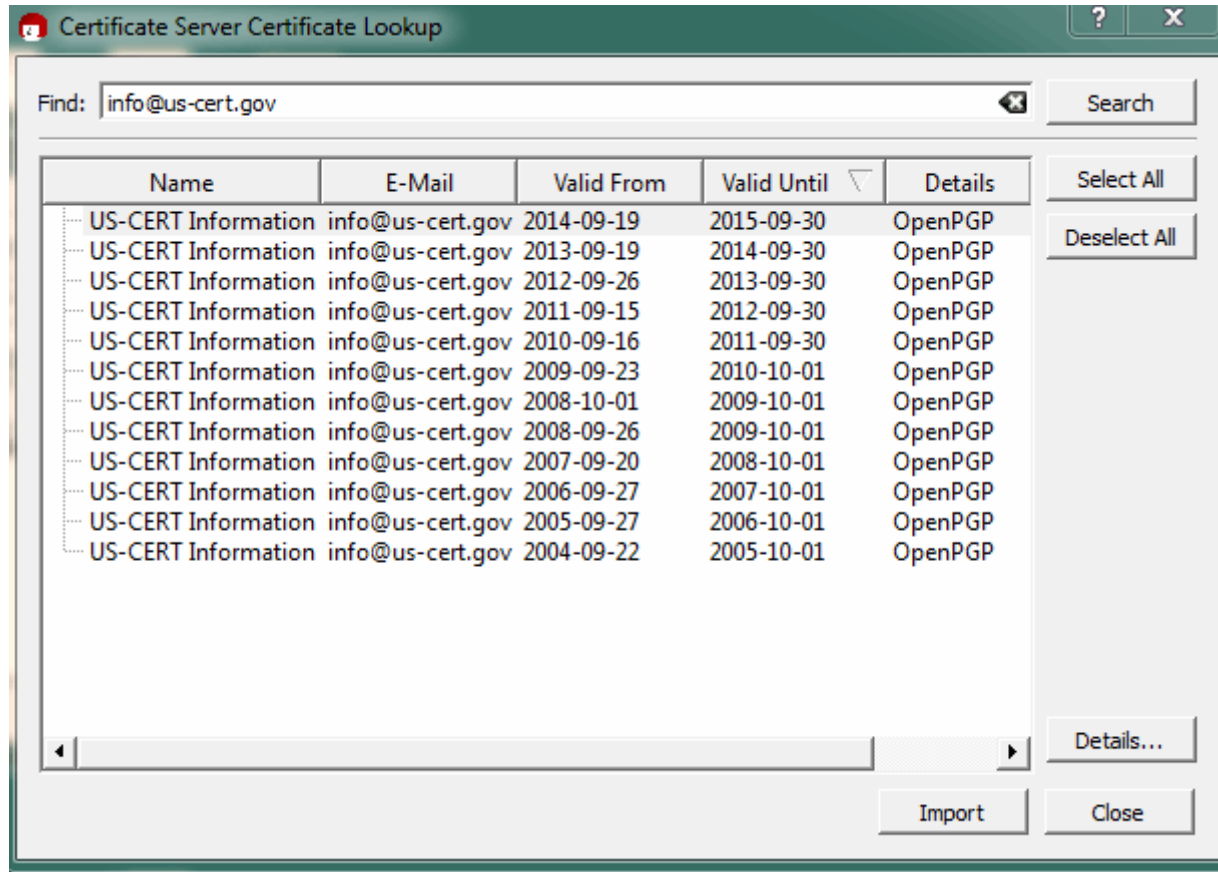
Note: You may also want to export a copy of your certificate (File → Export Certificate) for mailing directly to those who prefer not to use keyservers.

Windows-3:

Getting and Managing Other People's Public Keys

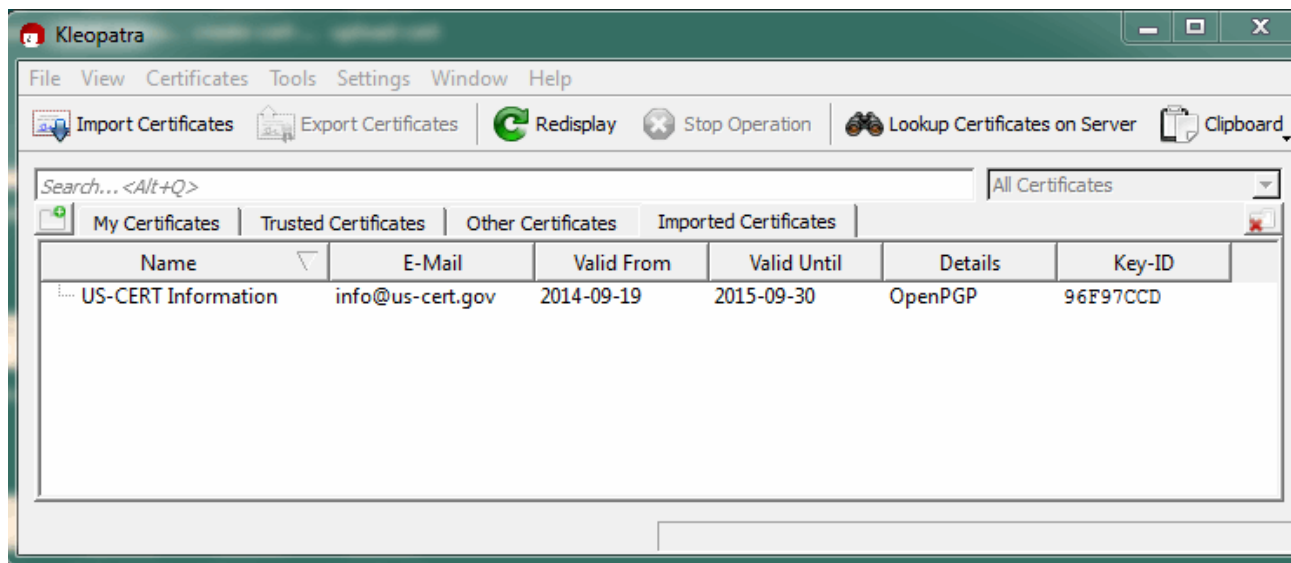
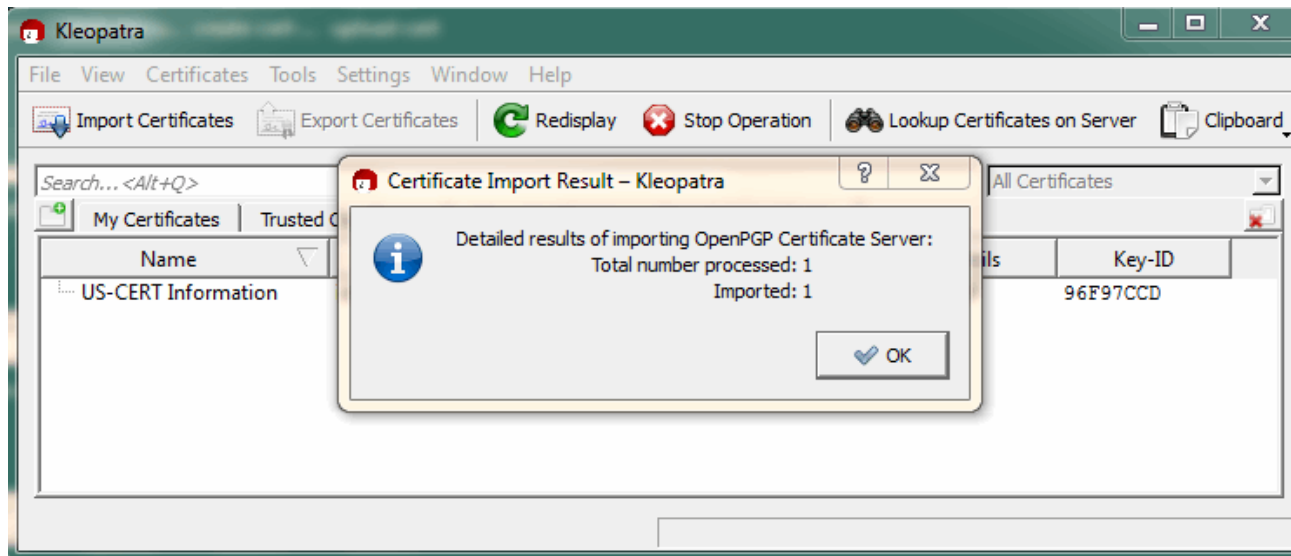
Look Up A Cert on the Public Keyservers

For example, let's find the one for info@us-cert.gov...



Suggestion: select the one that's currently valid! 😊

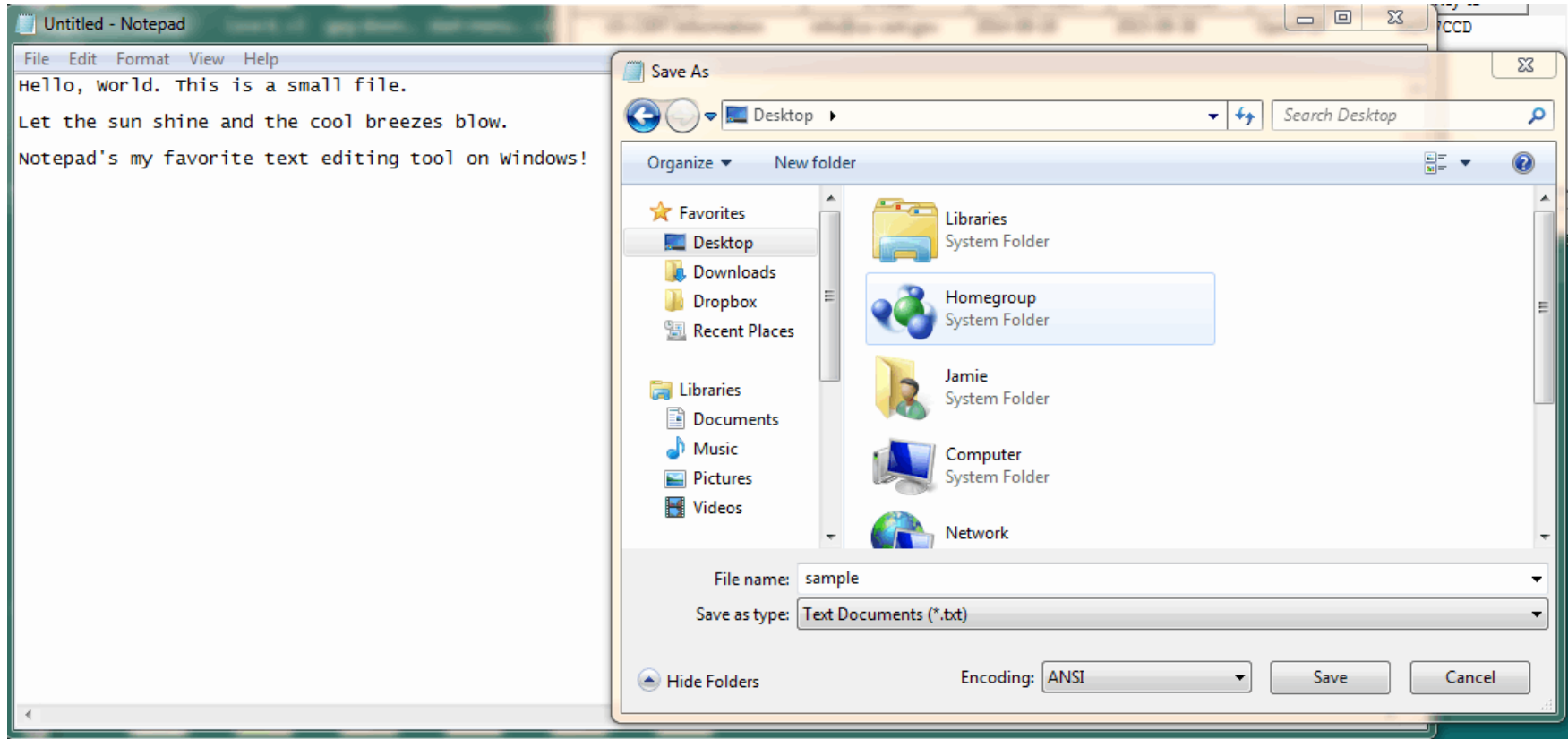
After Importing, You'll See...



Windows-4:

Signing A Message You've Made

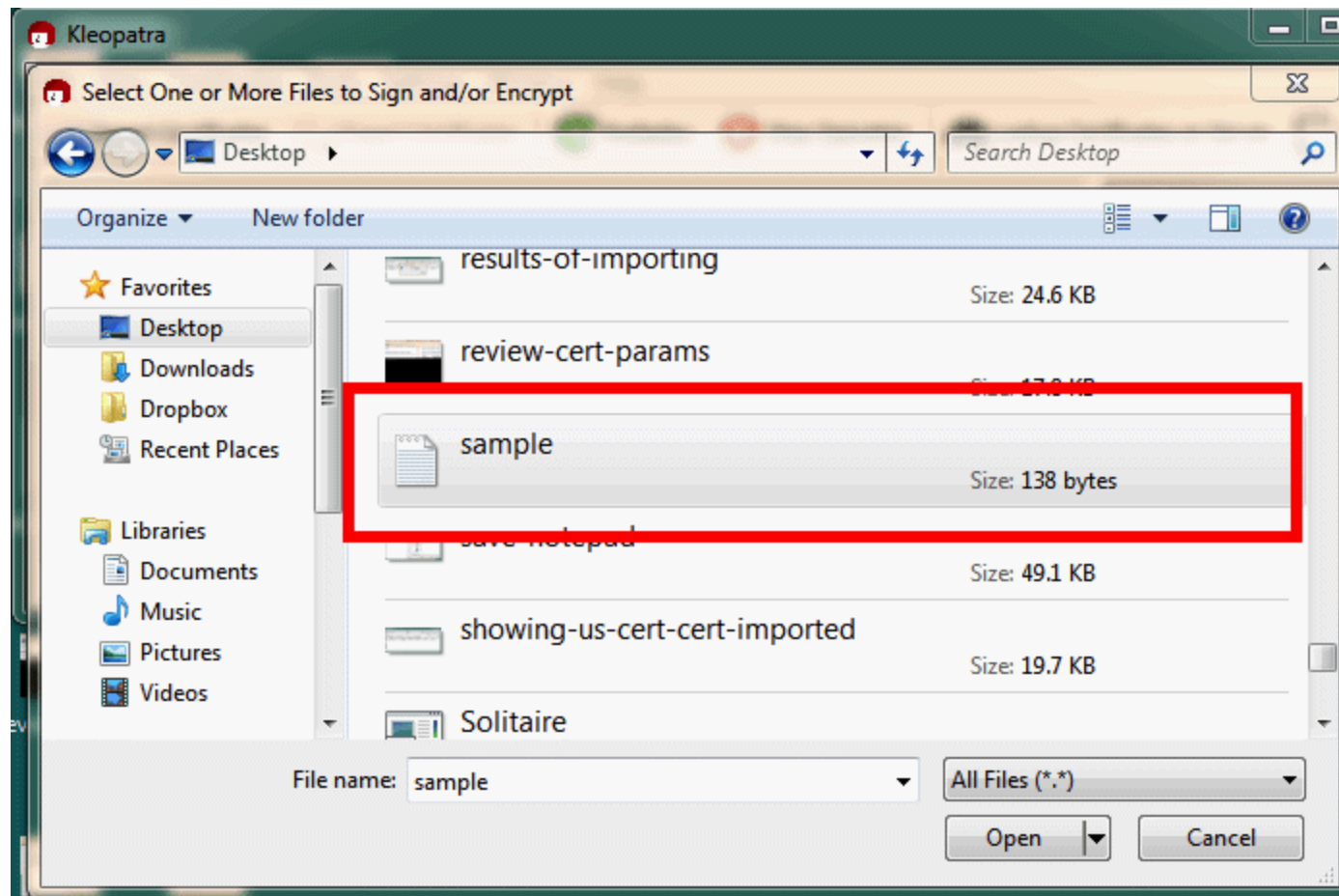
Create a Small Sample Message Using Notepad And Save It On Your Desktop (sample.txt) in ANSI Format



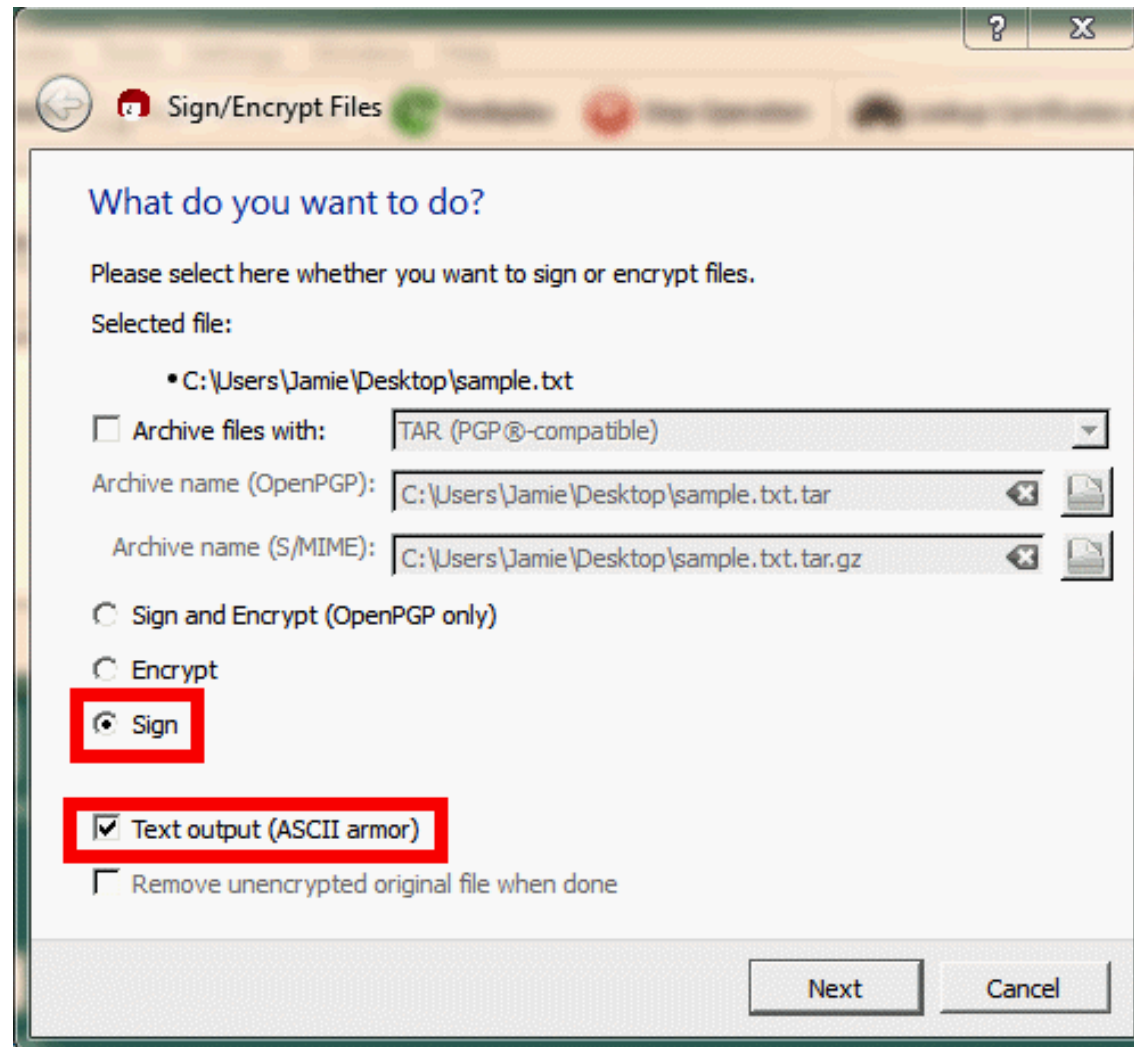
Note: If you don't normally use Notepad, you can find it under All Programs → Accessories folder → Notepad

Select That File For Signing

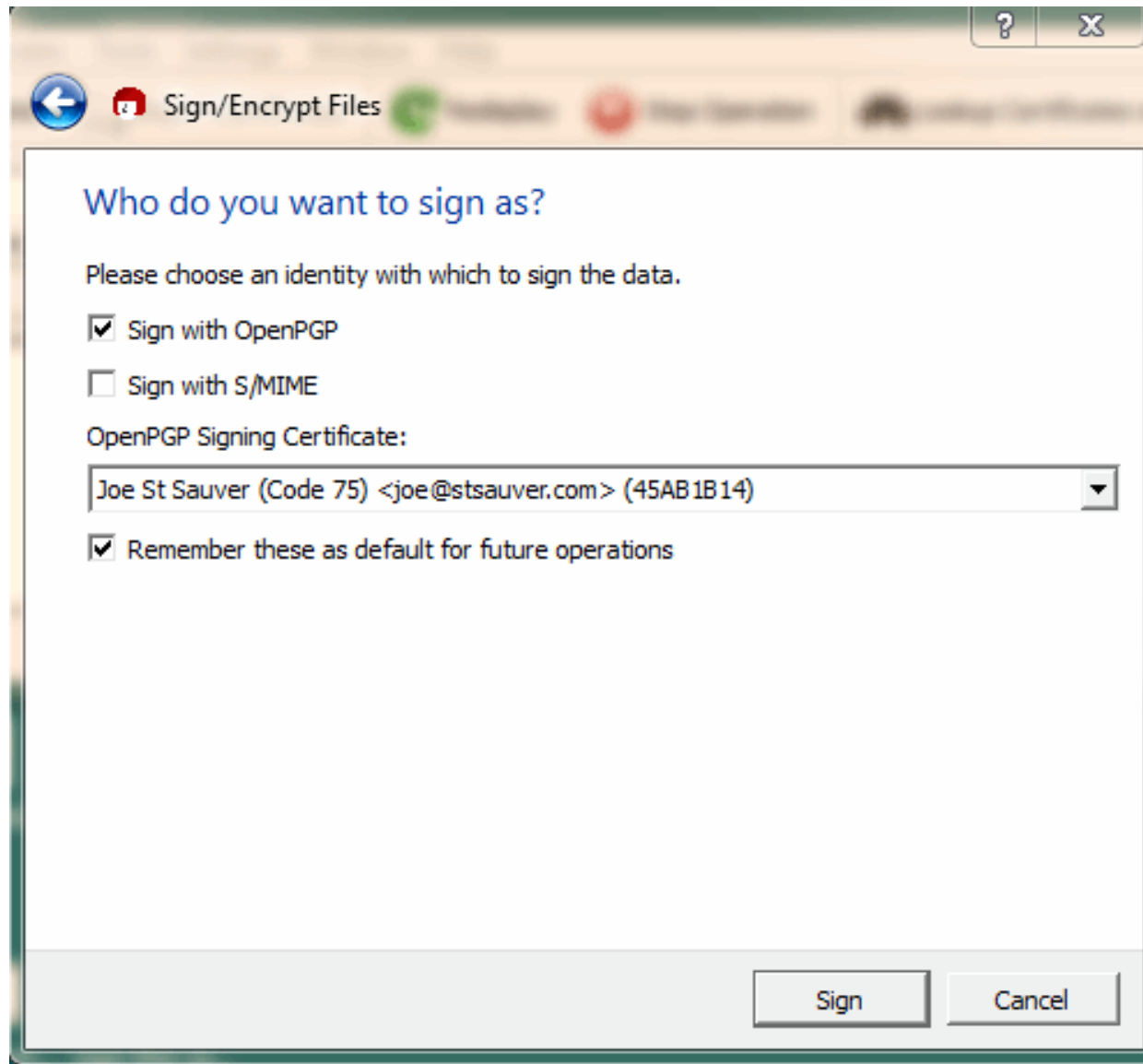
Go to File → Sign/Encrypt Files..., then select the sample.txt file from the Desktop (the .txt extension may not show by default)



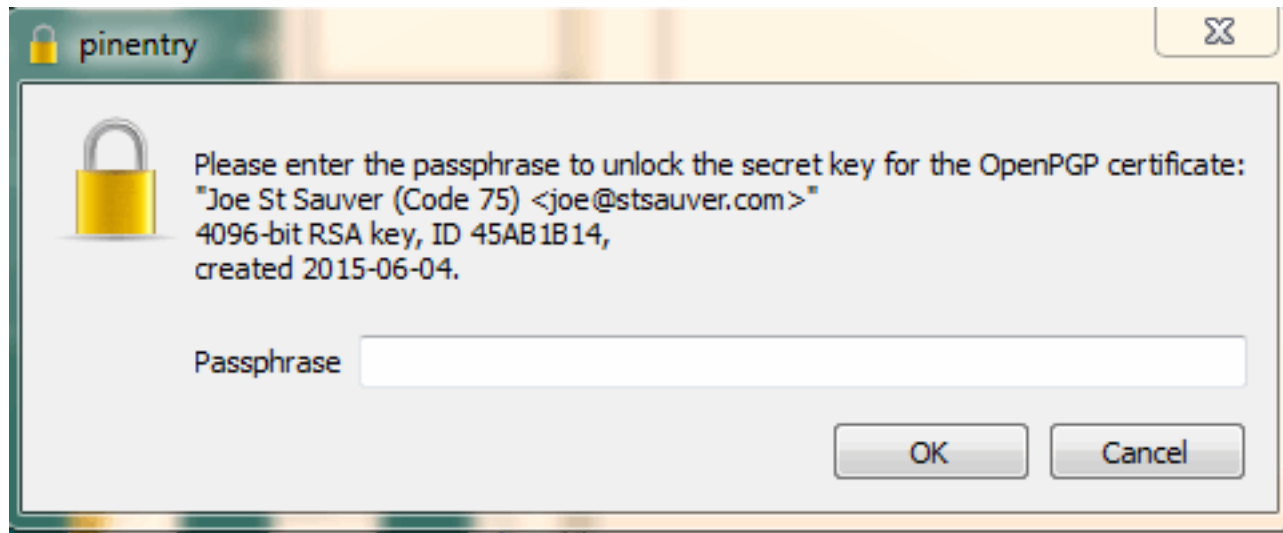
Sign The File



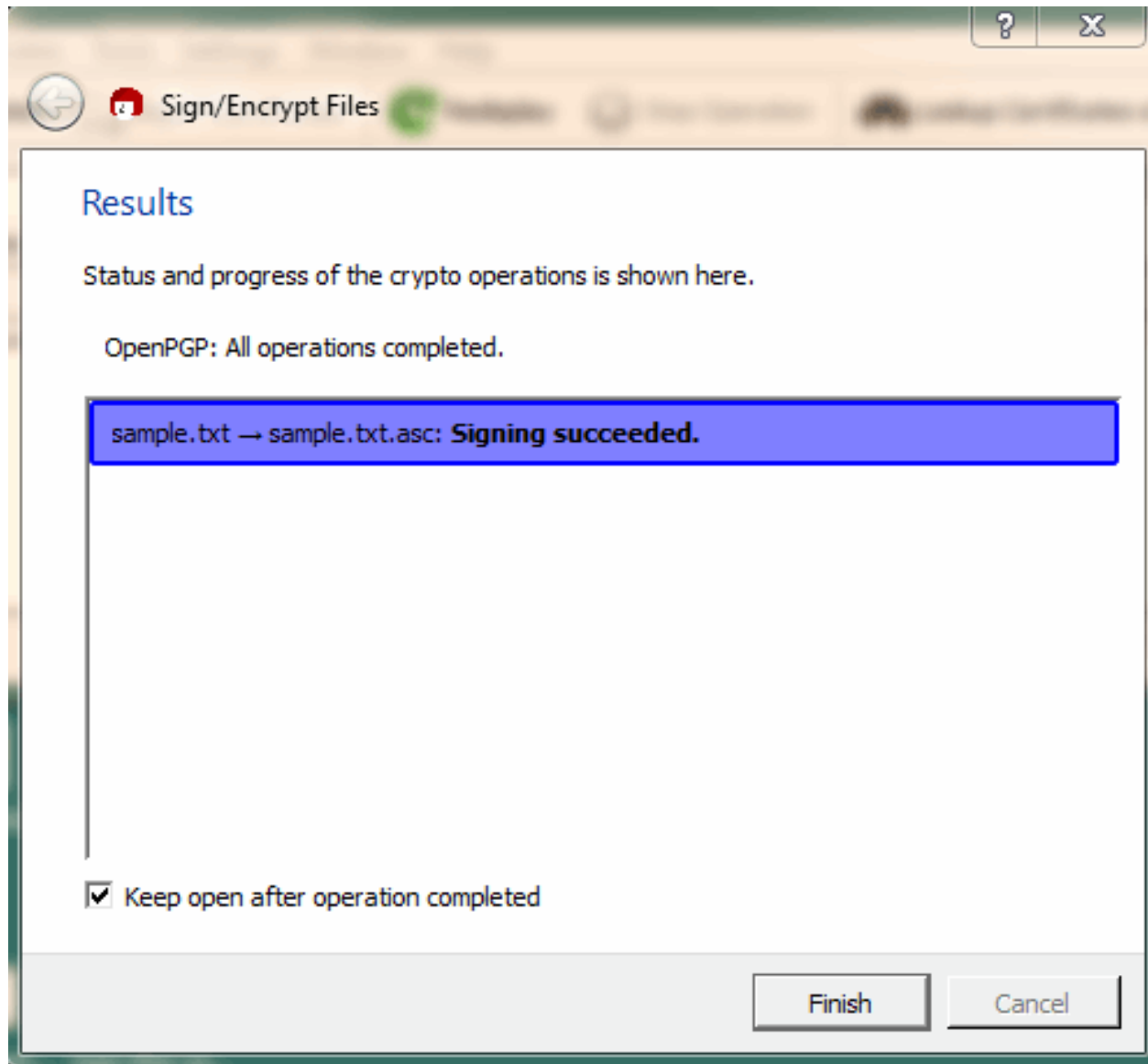
Pick HOW You'll Sign



Enter Your Passphrase



Signing Successful



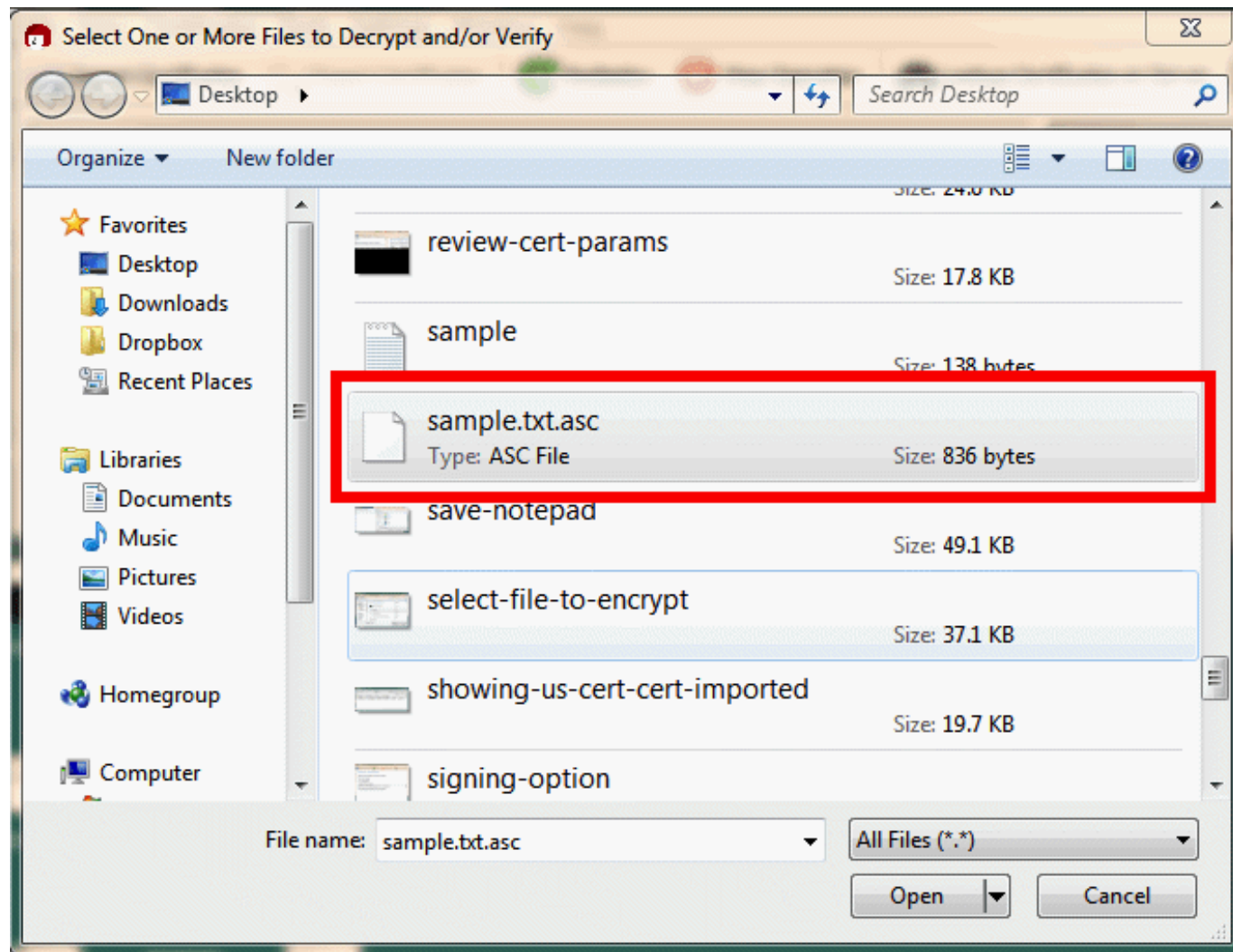
Windows-5:

**Validating A Signed Message
You've Received**

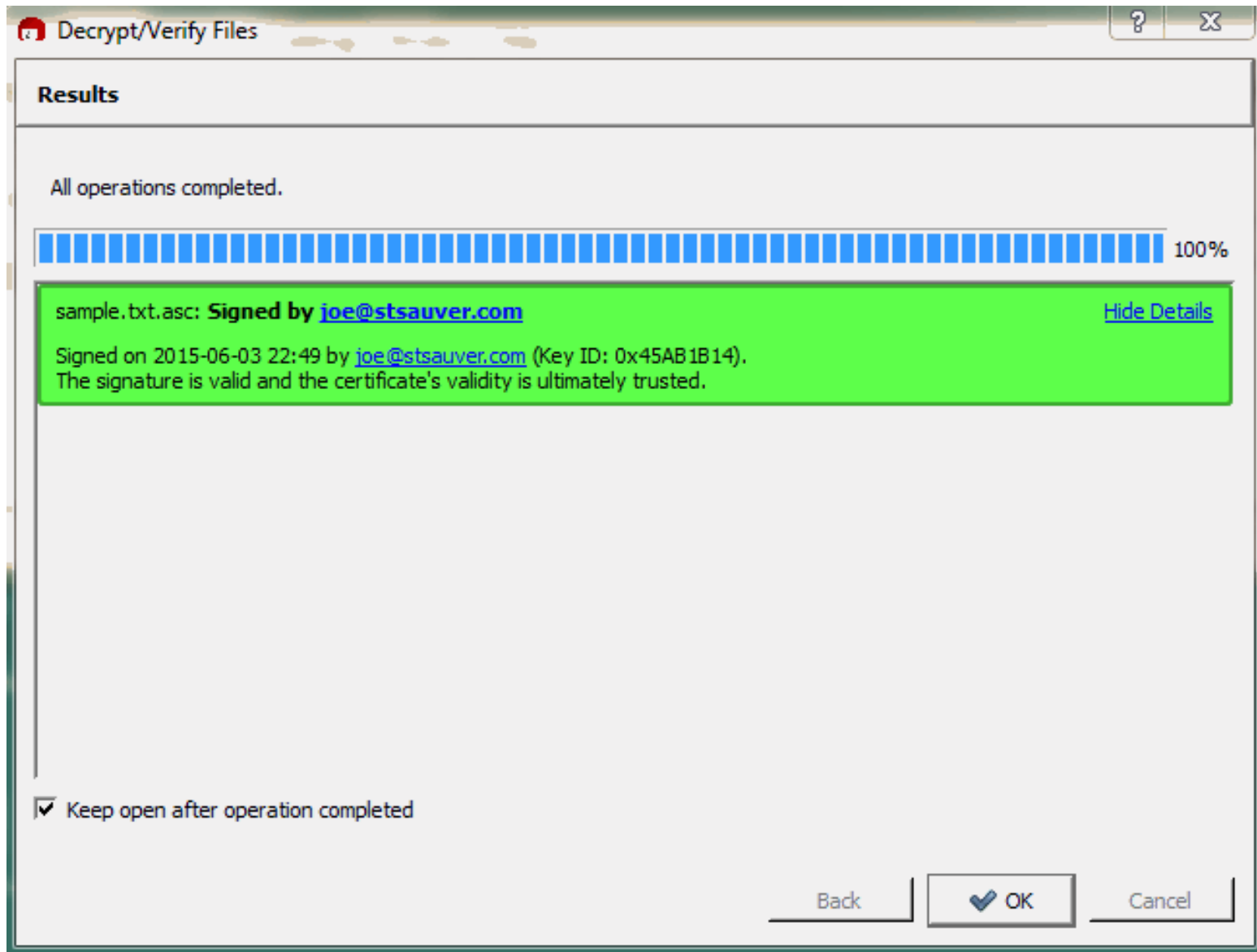
When You Receive A PGP-Signed Message...

- Save the message to a file.
- In Kleopatra, go to File → Decrypt/Verify Files... and select the file whose signature you want to verify
- If you don't have the key needed to validate that file, see the Windows-3 section above discussing how to get a user's public key (you probably WILL already have that key, at least if it's someone you routinely exchange PGP/GPG messages with)

Select A Signed File To Validate – Heck, Let's Validate The Sig on the Msg We Just Signed!



The Signature On Our File Is Valid



Different PGP/GPG Signature Structures

- One of PGP/GPG's challenges is the wealth of features it has accumulated over time. This is true for signatures as well as for other aspects of PGP/GPG.
- Signed documents can be in clearsign'd form, binary form, or ASCII armored form.
- Signatures can also be sent either as part of the file itself, or as a separate ("detached") signature.

If you're signing a text document, either will work, but if you're signing a binary file, the detached signature will ensure that the binary file isn't accidentally "munged" by the signing process.

(The downside of doing a detached signature is that both the original file AND the separate signature file need to get sent)

Files Damaged In Transit

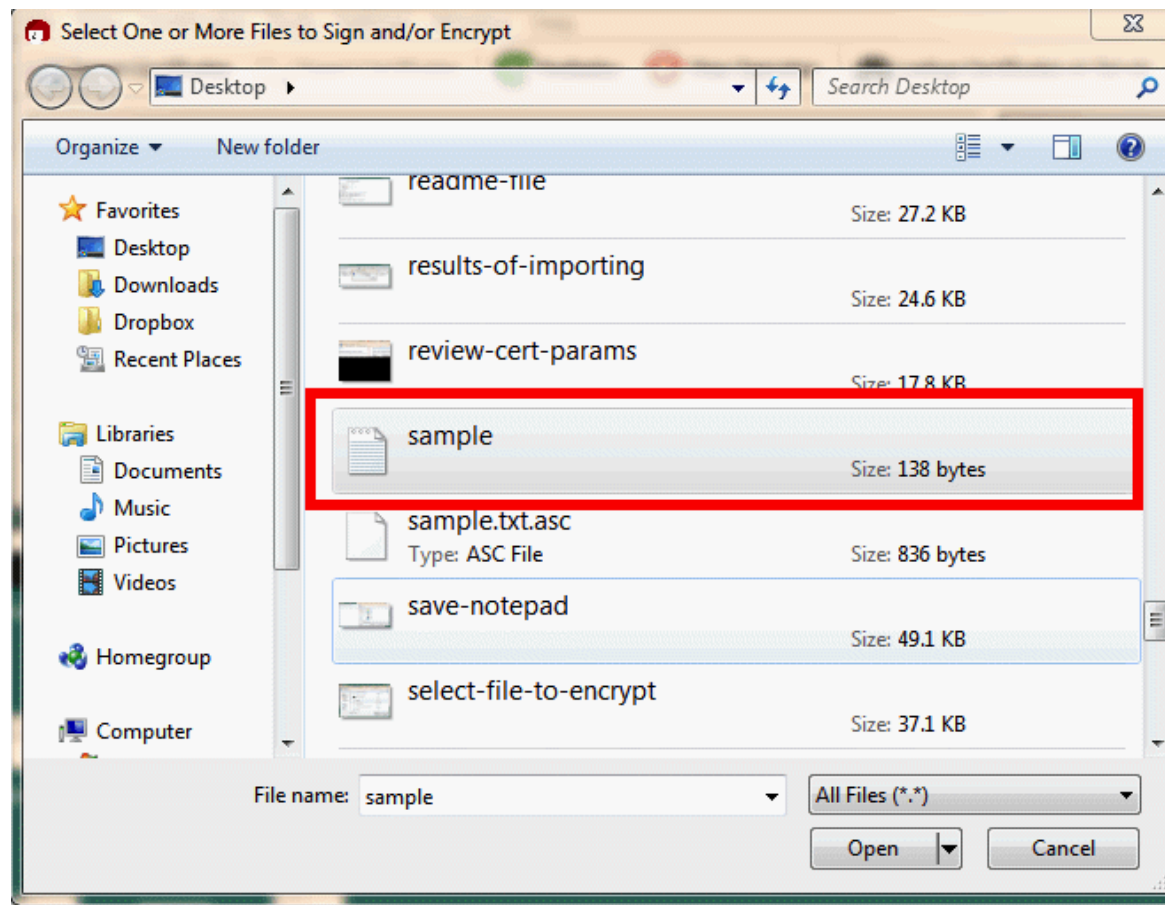
- Some mail clients may try to be "helpful" by doing things like wrapping lines or otherwise messing around with a message.
- In other cases, some sites may intentionally rewrite URLs in messages so that if any of those URLs ultimately "go rogue," security software can remotely break those dangerous links.
- **Sadly, ANY change to the body of a PGP/GPG-signed message, even just reformatting lines, will invalidate the signature associated with that message.**

Windows-6:

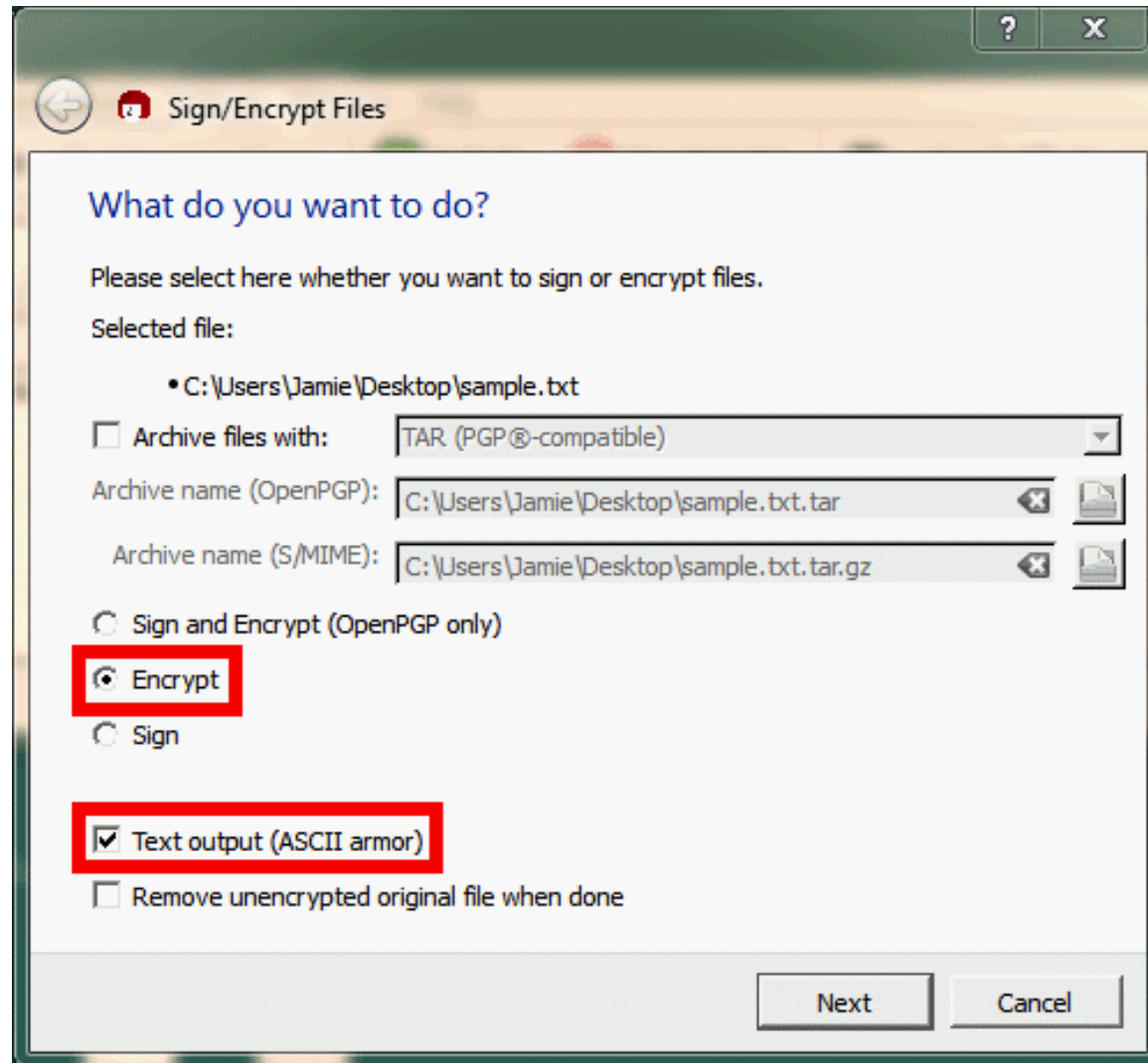
**Encrypting a Message To
Protect Against Eavesdropping**

Encrypting a File With Gpg4win

Start the encryption process by going to File → Encrypt/Sign... and picking the file you want to encrypt.



Select "Encrypt" and "Text Output"



BE CAREFUL! Do NOT accidentally click "remove unencrypted original file!"

On NOT Deleting Unencrypted Original Files

- The encryption panel in Gpg4win gives you the option of deleting your original unencrypted file after producing the encrypted version. While this is a tempting "convenience" option, I would encourage you to NOT routinely use it.
- My thinking:
 - a) Sometimes you simply want to keep an unencrypted copy of your original file
 - b) Other times you may *think* you're going to send an encrypted copy to yourself, but *forget to actually do so* – won't you be happy that you still have the unencrypted original?
 - c) You can manually delete the file with a file wiping program

Select A Recipient For The Encrypted File Then Hit the Green "Down Arrow" Add Button

Sign/Encrypt Files

For whom do you want to encrypt?

Please select for whom you want the files to be encrypted. Do not forget to pick one of your own certificates.

Search... All Certificates

Name	E-Mail	Valid From	Valid Until
US-CERT Information	info@us-cert.gov	2014-09-19	2015-09-30
Joe St Sauver (Code 75)	joe@stsauver.com	2015-06-03	2017-06-03

▼ Add ▲ Remove

Name	E-Mail	Valid From	Valid Until	De
------	--------	------------	-------------	----

Encrypt Cancel

For Example, We Chose US-CERT...

Sign/Encrypt Files

For whom do you want to encrypt?

Please select for whom you want the files to be encrypted. Do not forget to pick one of your own certificates.

Search... All Certificates

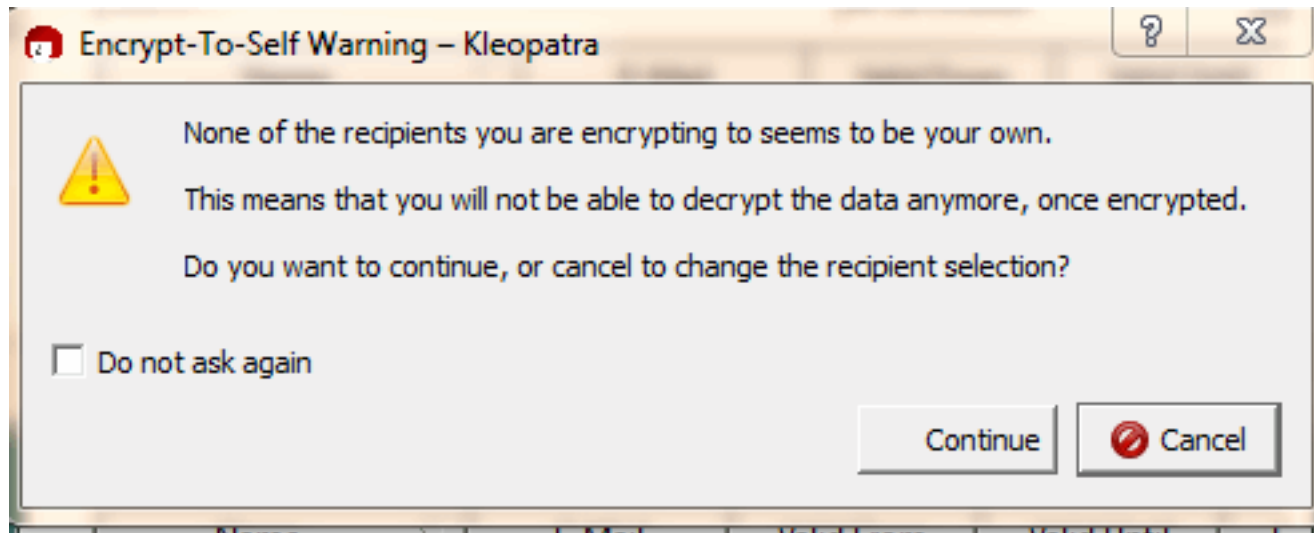
Name	E-Mail	Valid From	Valid Until
US-CERT Information	info@us-cert.gov	2014-09-19	2015-09-30
Joe St Sauver (Code 75)	joe@stsauver.com	2015-06-03	2017-06-03

▼ Add ▲ Remove

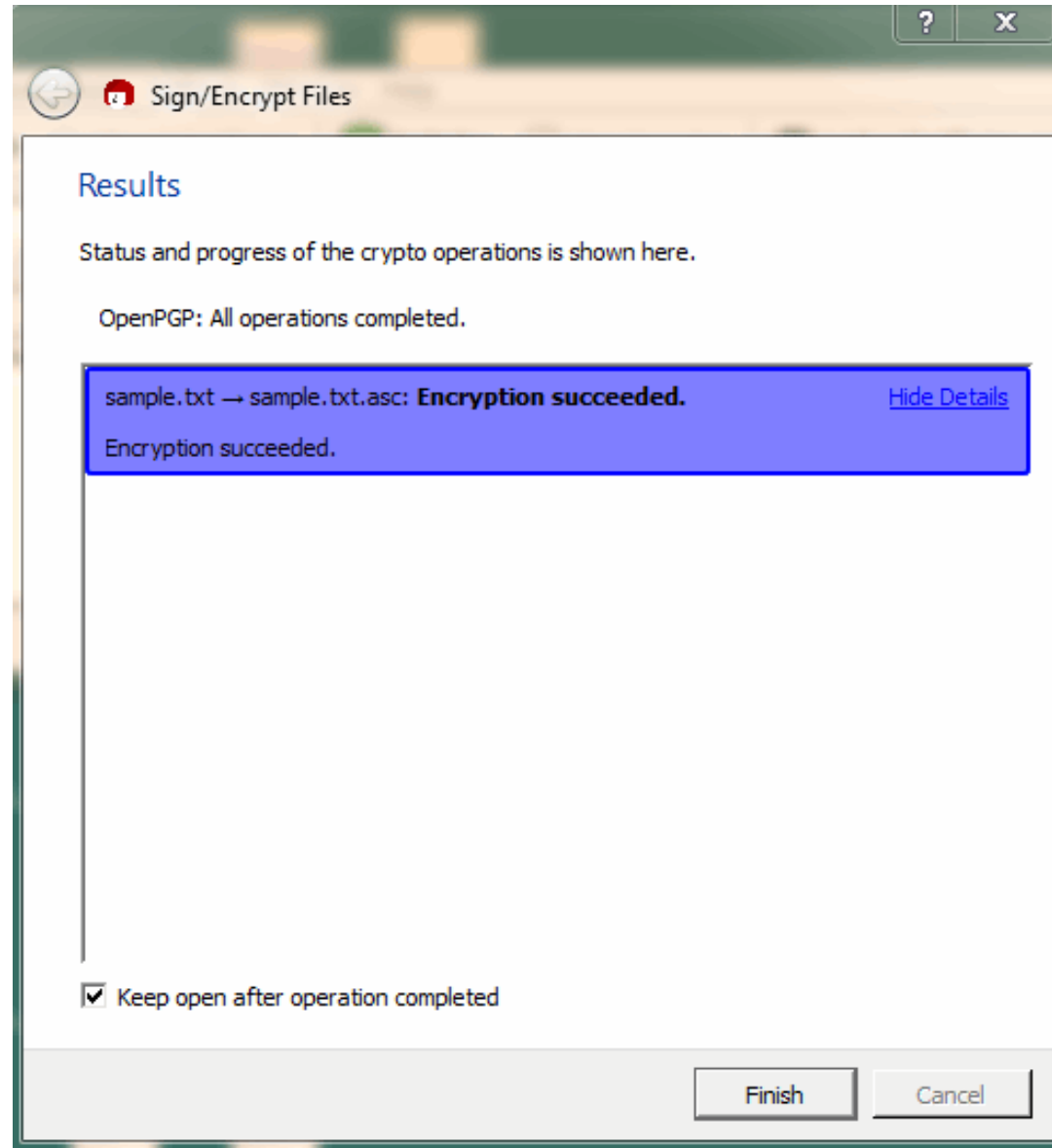
Name	E-Mail	Valid From	Valid Until	
US-CERT Information	info@us-cert.gov	2014-09-19	2015-09-30	Open

Encrypt Cancel

Since We DIDN'T Also Add Ourselves, We're Going to Get A Warning



Wingpg4 Verifies That The Encryption's Done



Send That Encrypted File To Your Recipient

- The encrypted file temp.txt.asc can then be sent to your recipient by cutting and pasting it into a regular email message
- **NOTES:** Be SURE you get the right file, e.g., the encrypted one!
- Encrypted files about sensitive topics should NOT have revealing email **subject lines** (remember, those are NOT encrypted)
- If you encrypt one file to multiple people, each person will be able to ascertain the identities of all the other recipients also able to read that file; in some circumstances, this may be sensitive information. You can deal with this by encrypting your message for just one user at a time.

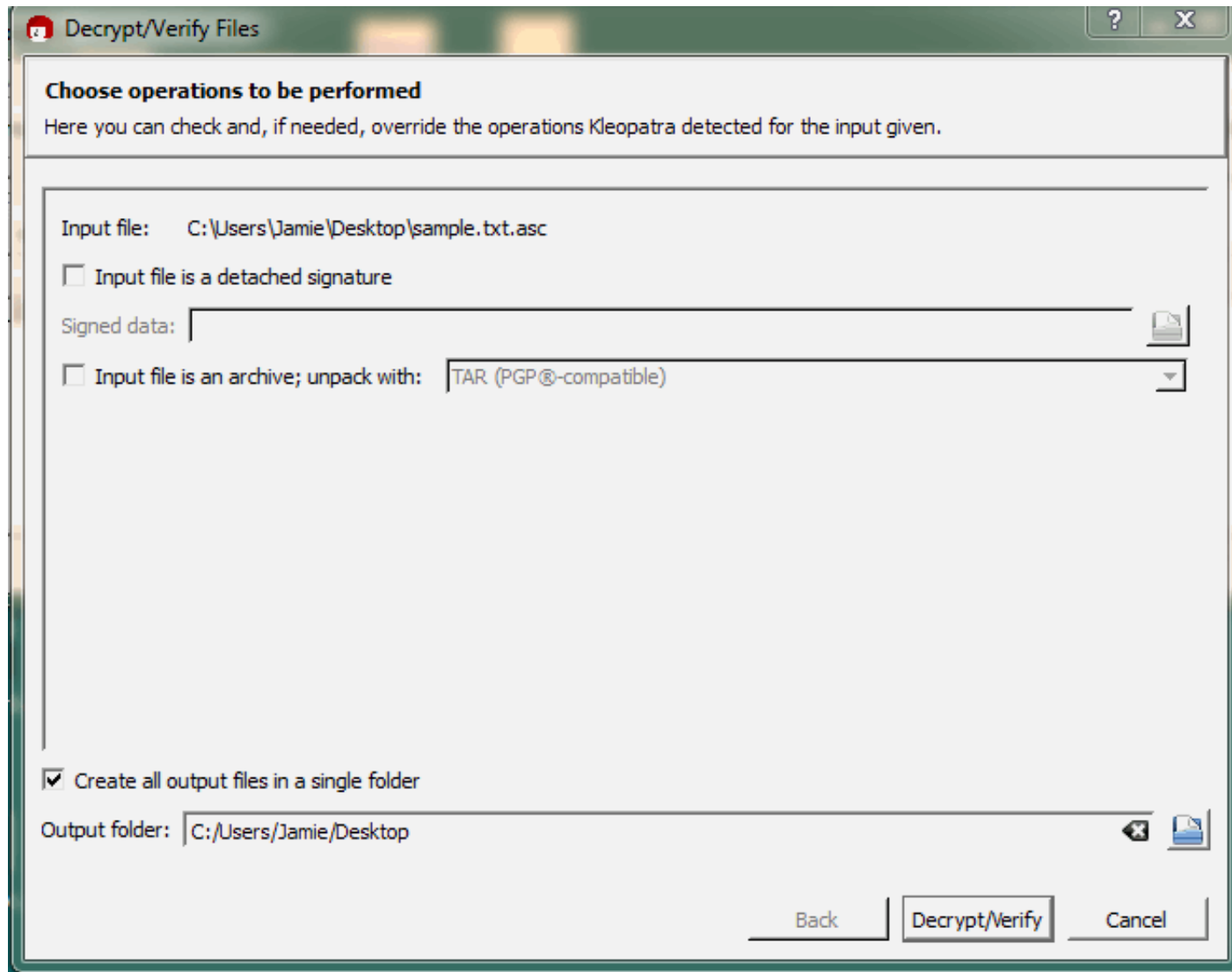
Windows-7:

**Decrypting an Encrypted Message
You've Just Received**

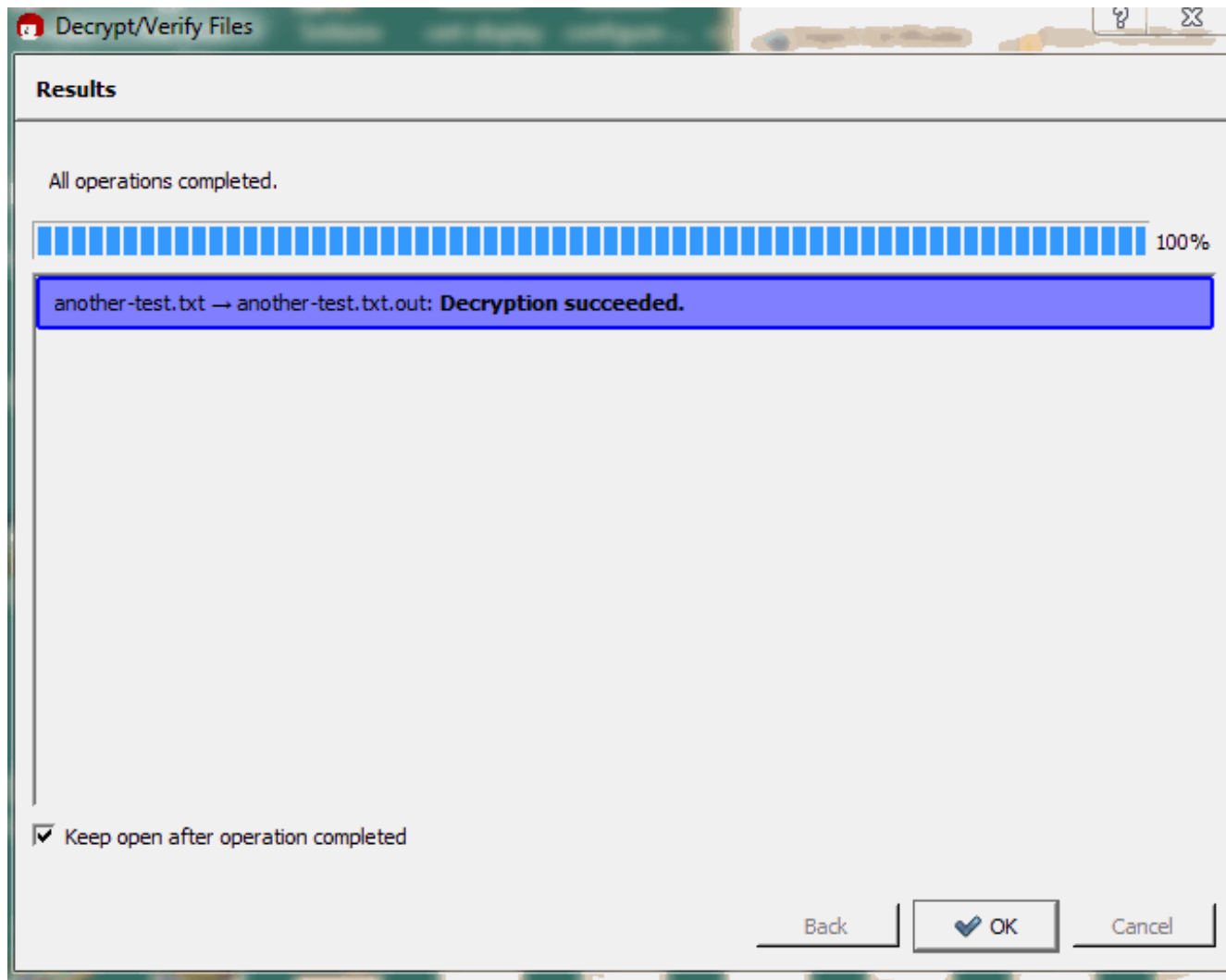
To Decrypt an Encrypted Message You've Received

- Copy and paste or save the encrypted message to a file.
- In Kleopatra, go to File → Decrypt/Verify
- Choose the file you want to decrypt

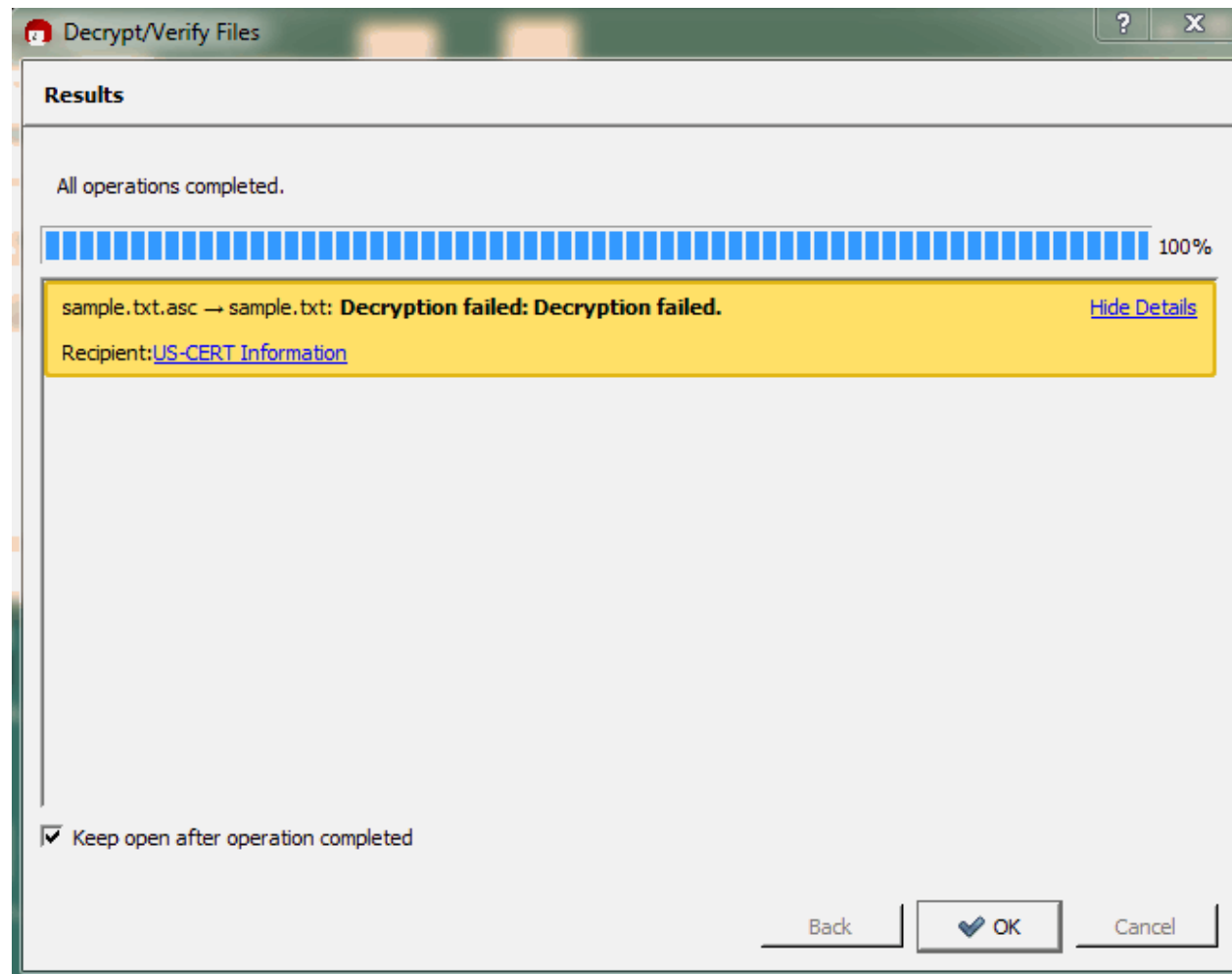
Decrypting...



Successful Decryption!



What You'll See If You CAN'T Decrypt (Since I'm Not US-CERT, And This Message Was Encrypted ONLY For Them With Their Key, This Makes Sense, Right?)

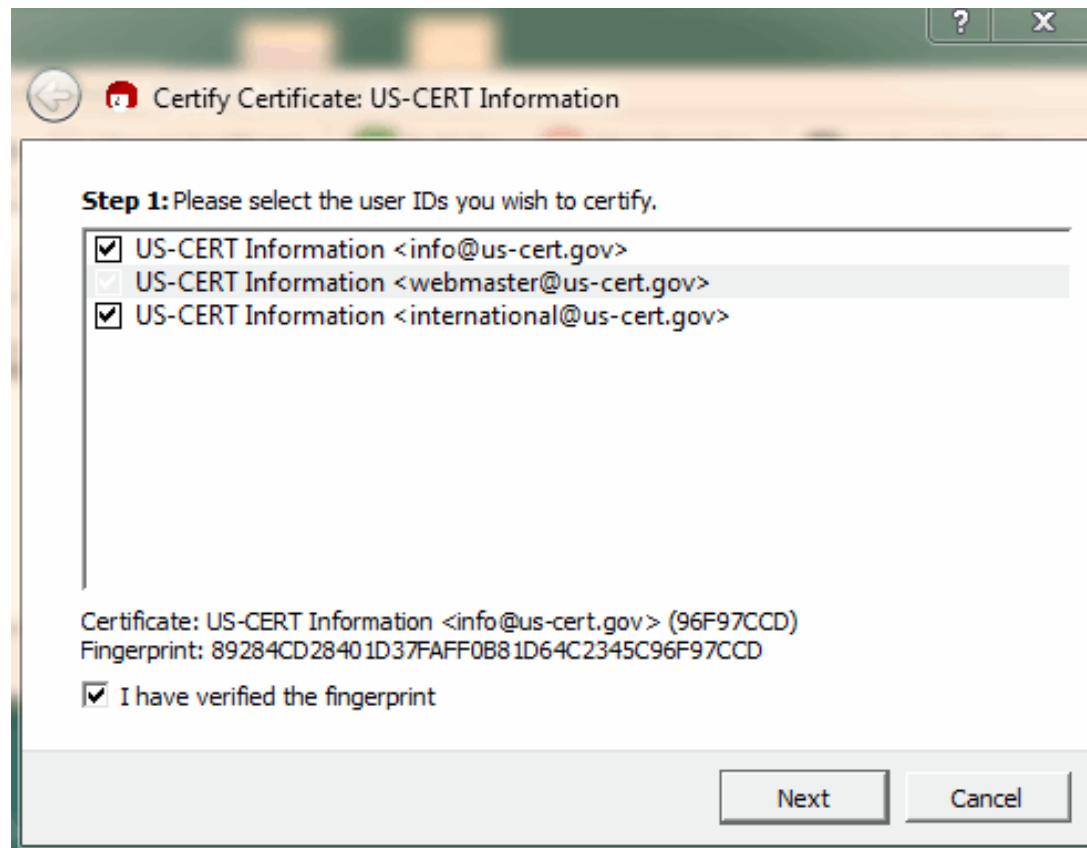


Windows-8:

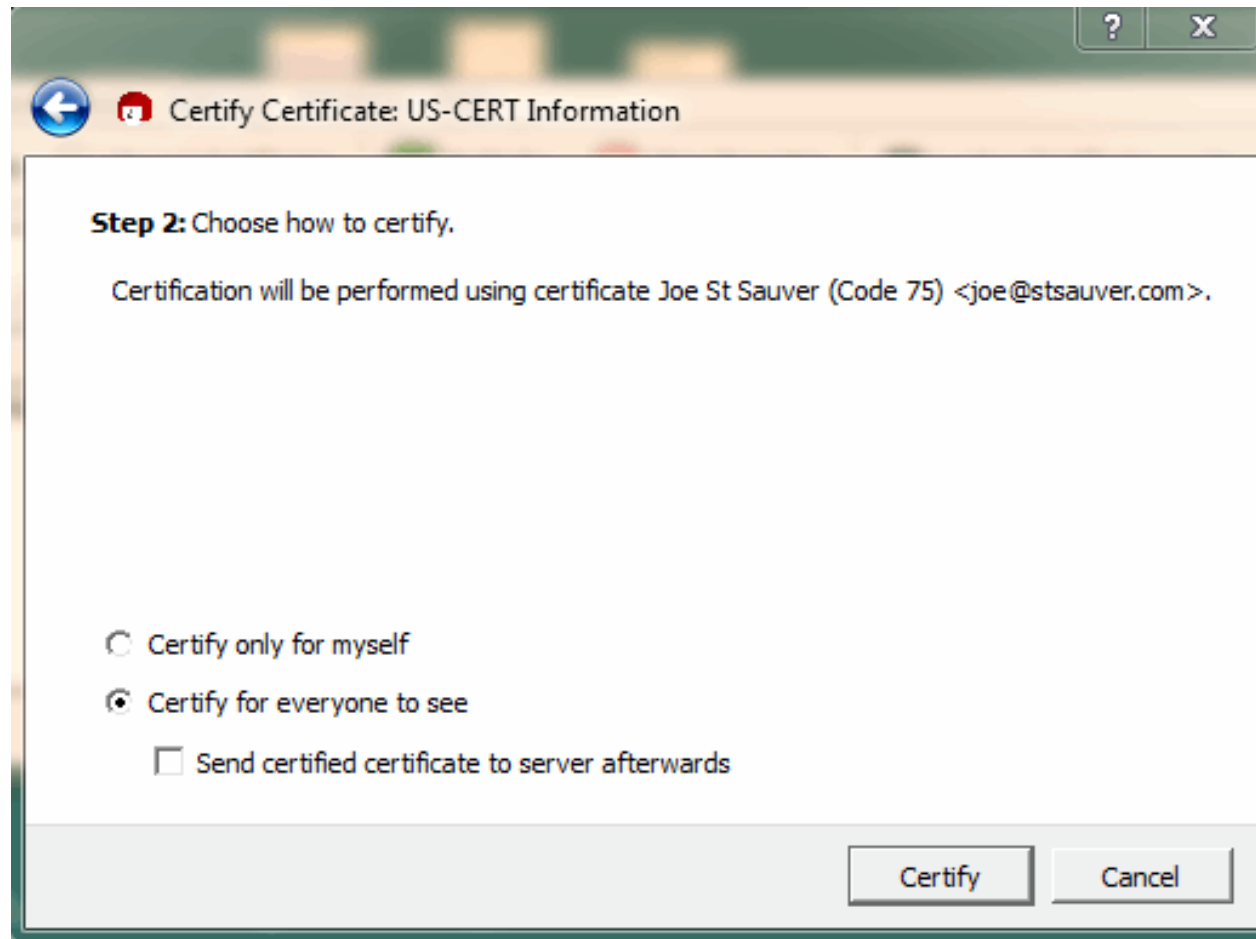
Signing Another User's Public Key

To Sign Another User's Key (After You've Verified Their Passport & Confirmed Their Key Fingerprint)...

Select the public key you want to sign from the Kleopatra's List of "Other" certificates, then go to Kleopatra's Certificates Menu → Certify Certificates and pick the relevant IDs

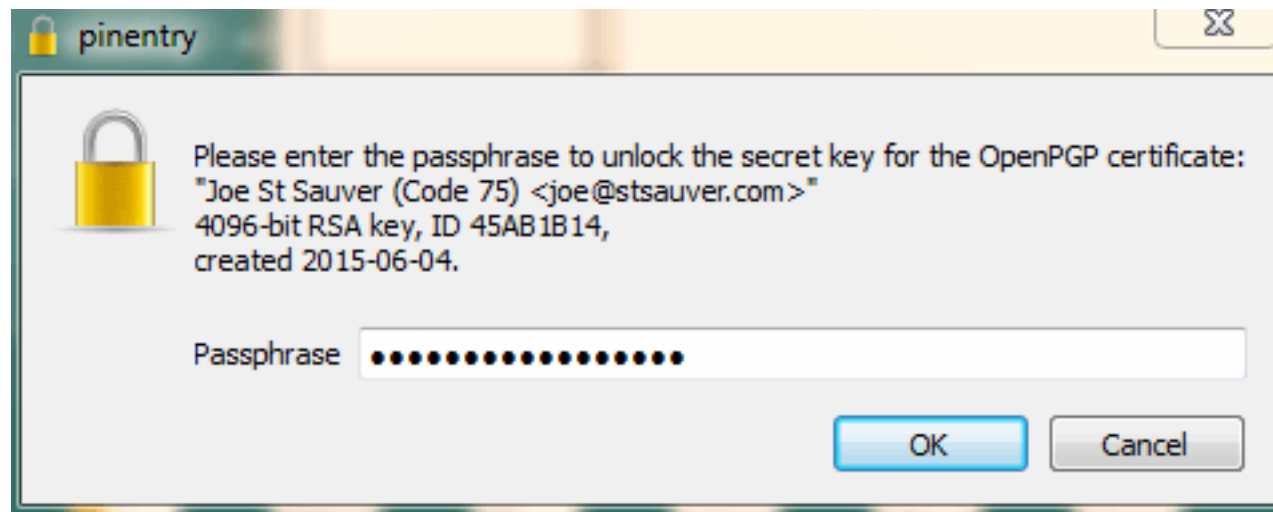


Confirm That You Want To Publicly Sign It...



Do NOT click "send certified certificate to server afterwards"

Enter Your Passphrase



Send The Signed Key Back To Its Owner

- Select the key you just signed
- Click "Export Key"
- Save that key to a file
- Send that file to its owner by email for their use (or for THEM to publish to a public key server, if they want to do so)
- Is this a bit of a hassle? Yes. Is it the courteous way to sign a key? Yes.

Windows-9:

**Importing Your Own Key
After It Has Been Signed**

Assume Someone's Just Sent You Your Public Key, Signed By Them

- Check your notes: is this someone who properly validated my ID and key fingerprint somewhere? If so, save the signed key to a file. If not maybe check with the user and figure out what's going on?
- In the Kleopatra certificate viewer, use the Import Certificates button to select the file you just created (containing the newly signed key) and add it to your key ring.
- Optionally, export that signed file to a key server by going to File → Export Certificates to Server...

**ONLY EXPORT YOUR OWN PUBLIC KEYS TO A KEYSERVER;
DO NOT PRESUME TO MAKE THIS IRREVOCABLE CHOICE FOR
OTHERS!**

Windows-10:

PRACTICE!

Windows Users: Practice For A Few Minutes!

- You've now got all the skills you need to do simple PGP/GPG messages.
- Pair up with another user from class, and do the following:
 - a) create a key pair for yourself if you haven't already done so
 - b) get a copy of your partner's public key
 - c) send a signed message to your partner
 - d) send an encrypted message to your partner
 - e) verify the signature of the message you received from your partner
 - f) decrypt the message you received from your partner
 - g) if so inclined, sign your partner's key and send it back to them

GPG CONVENIENCE TOOLS

Integrate GPG With Thunderbird Using Enigmail

- The approach we've shown up until now is (candidly) pretty clunky, often involving copying and pasting stuff. While that works fine for an occasional message, you probably wouldn't want to have to do it for a hundred messages a day.
- If you're going to be routinely using GPG all the time, you probably want it to be closely integrated with a point-and-click email client such as Mozilla Thunderbird.
- The best way to integrate GPG with Thunderbird is by installing the free Enigmail plugin. It works on Macs, PCs, Linux, etc.
- It's available from <https://www.enigmail.net/download/index.php> or in Thunderbird go to Tools → Addons and search for enigmail
- If you don't already have Thunderbird installed, you can get it from <https://www.mozilla.org/en-US/thunderbird/>

Thunderbird Configuration Caution #1

- Thunderbird supports both POP and IMAP connections.
- Thunderbird will attempt to heuristically configure your connection the first time you run it.
- **BE CAREFUL. Do NOT configure Thunderbird to use POP** to access your existing email account! If you do, you will download all your existing email to your laptop and it may be deleted from your server!
- **Use IMAP instead! Do NOT use POP!**

GMail and IMAP

Google Search Gmail Help

Gmail Help

Use email clients with IMAP or POP > IMAP Basics

Get started with IMAP and POP3

- What is POP and IMAP?
- How much does POP and IMAP cost?
- What's the difference between POP and IMAP?
- How will using a mail client affect automatic replies?

Select an option below for instructions on how to enable POP or IMAP

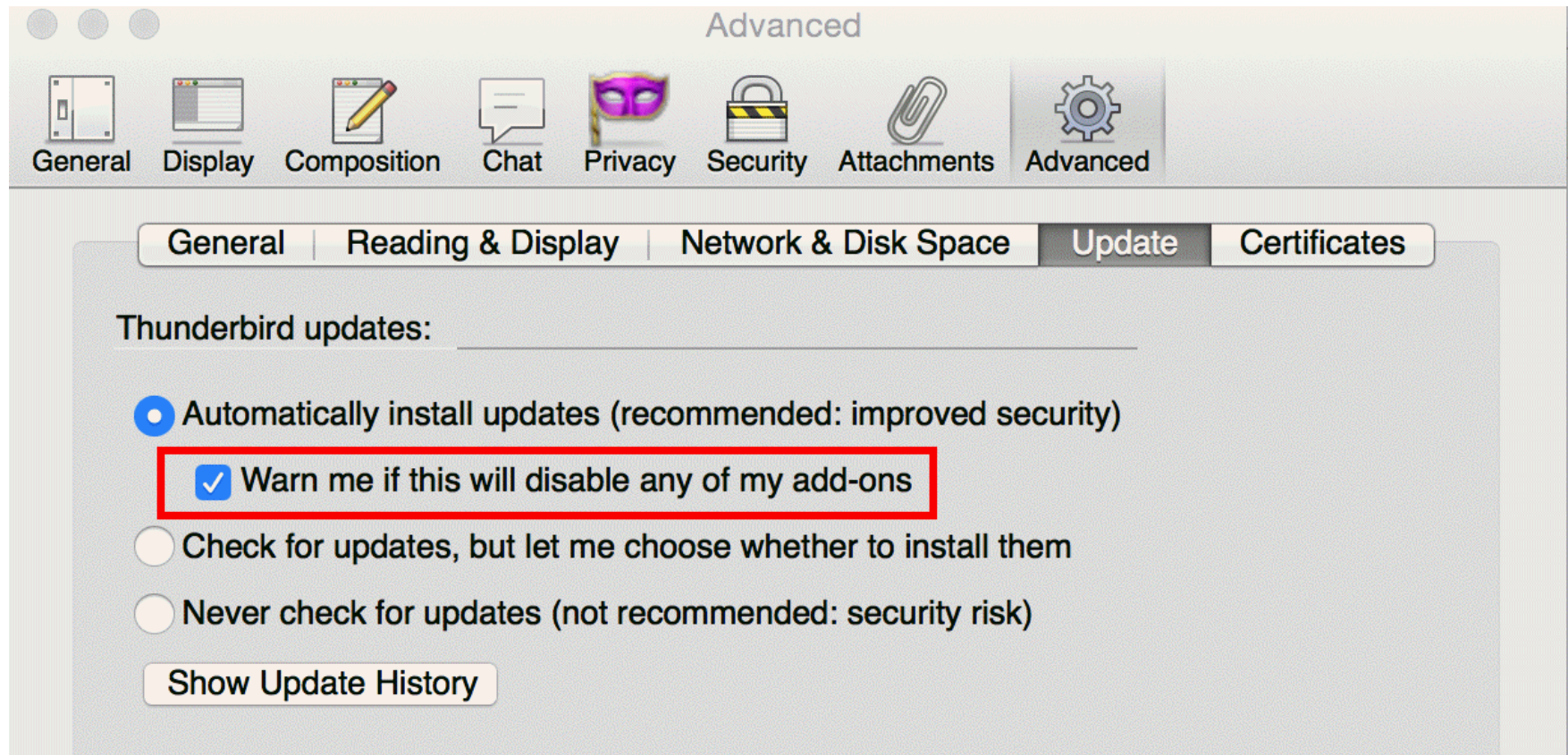
☒ I want to enable IMAP ONLY USE IMAP!

☐ I want to enable POP

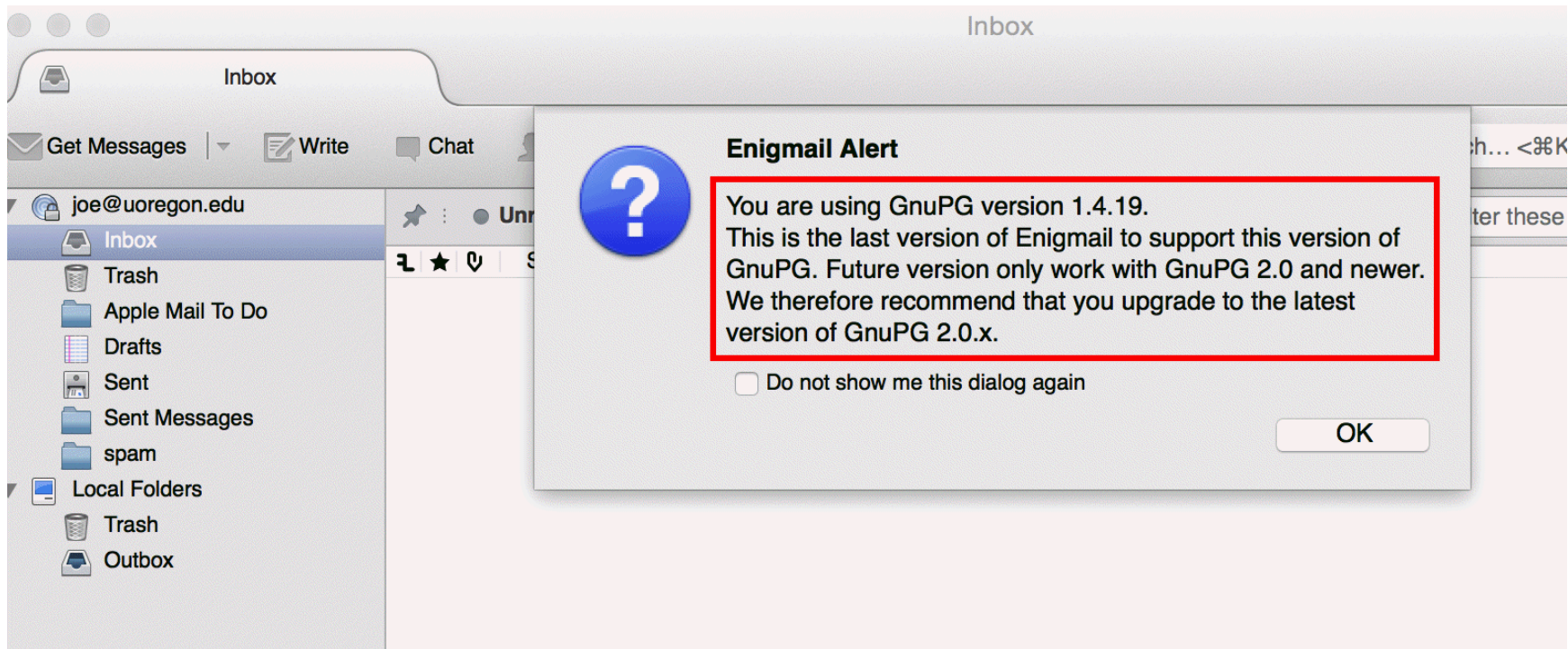
Caution #2: Enigmail and Thunderbird Updates

- From time-to-time, the Thunderbird project may produce updated versions of Thunderbird. When that happens, the Enigmail plugin may continue to be compatible, or it may need to be tweaked and updated in order to continue to interoperate. **If you routinely use and heavily rely on Enigmail, do NOT update Thunderbird unless you're SURE Enigmail is compatible with the new version.**
- Normally you will be automatically prevented from "shooting yourself in the foot" this way, but **sometimes folks will override the compatibility check and install an incompatible new version without pondering the implications of their decision.**
- Don't do that. Really. Just don't.

The Critical Check In Thunderbird Advanced Prefs



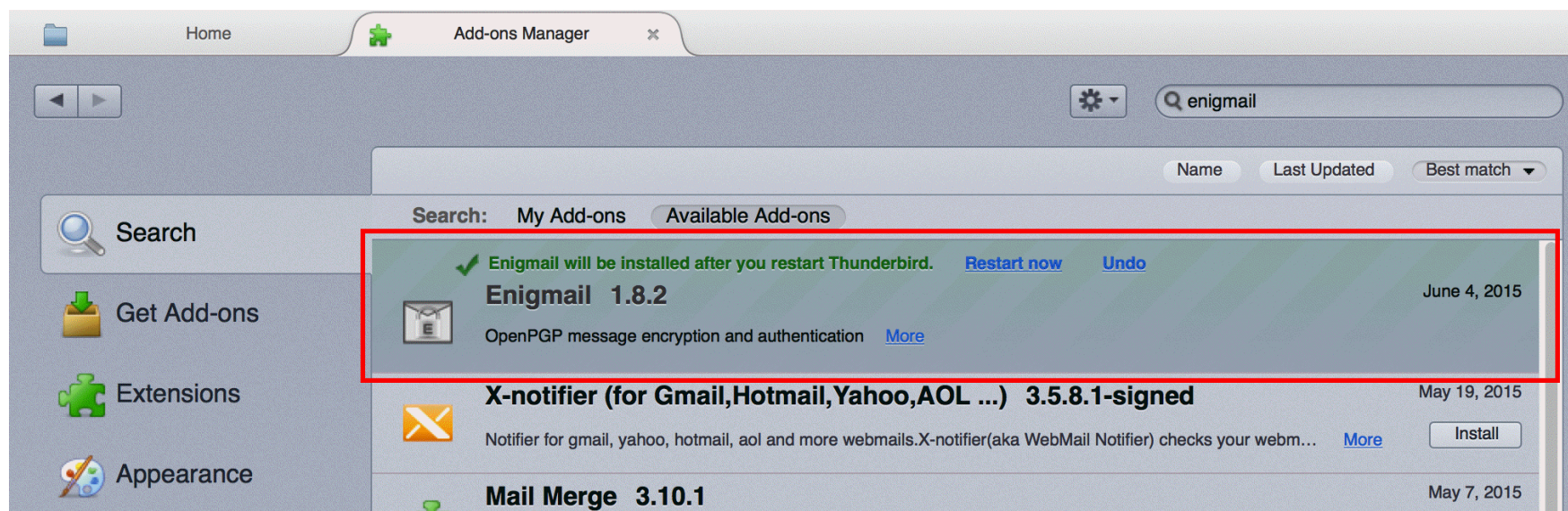
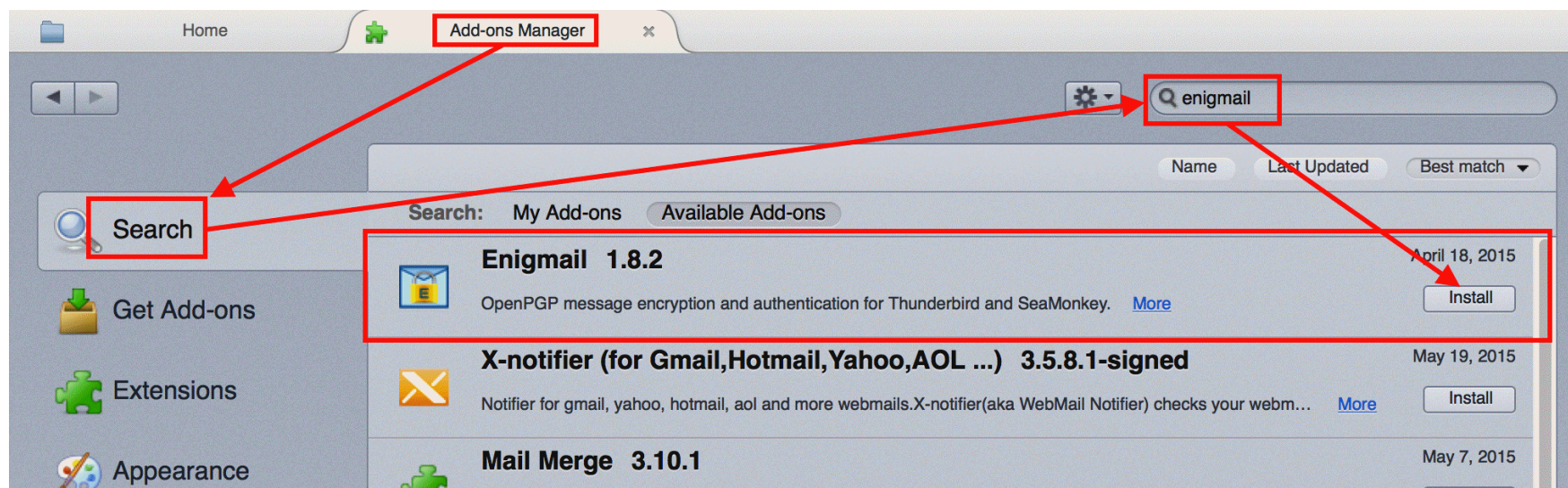
Caution #3: Last Version to Support Classic GPG



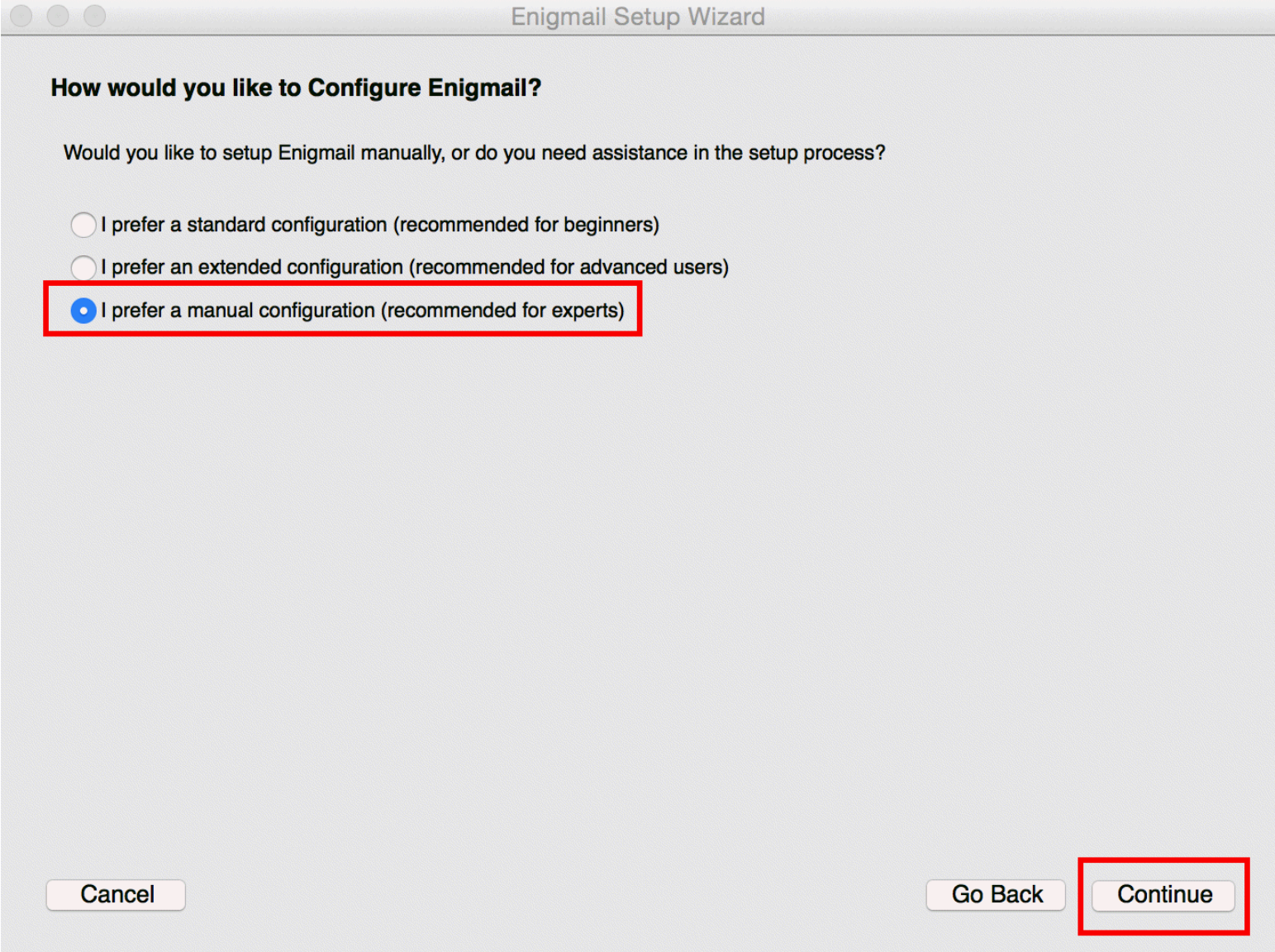
Note: one Enigmail option is to install GNU Privacy Guard 2.x along side GNU Privacy Guard 1.4.19.

Let's proceed, notwithstanding this...

Installing Enigmail From Within Thunderbird



There's a Setup Wizard, But You Don't Need It



The image shows a screenshot of a macOS-style dialog box titled "Enigmail Setup Wizard". The dialog has a light gray background and a title bar with three window control buttons (red, yellow, green) on the top left. The main content area contains the following text and options:

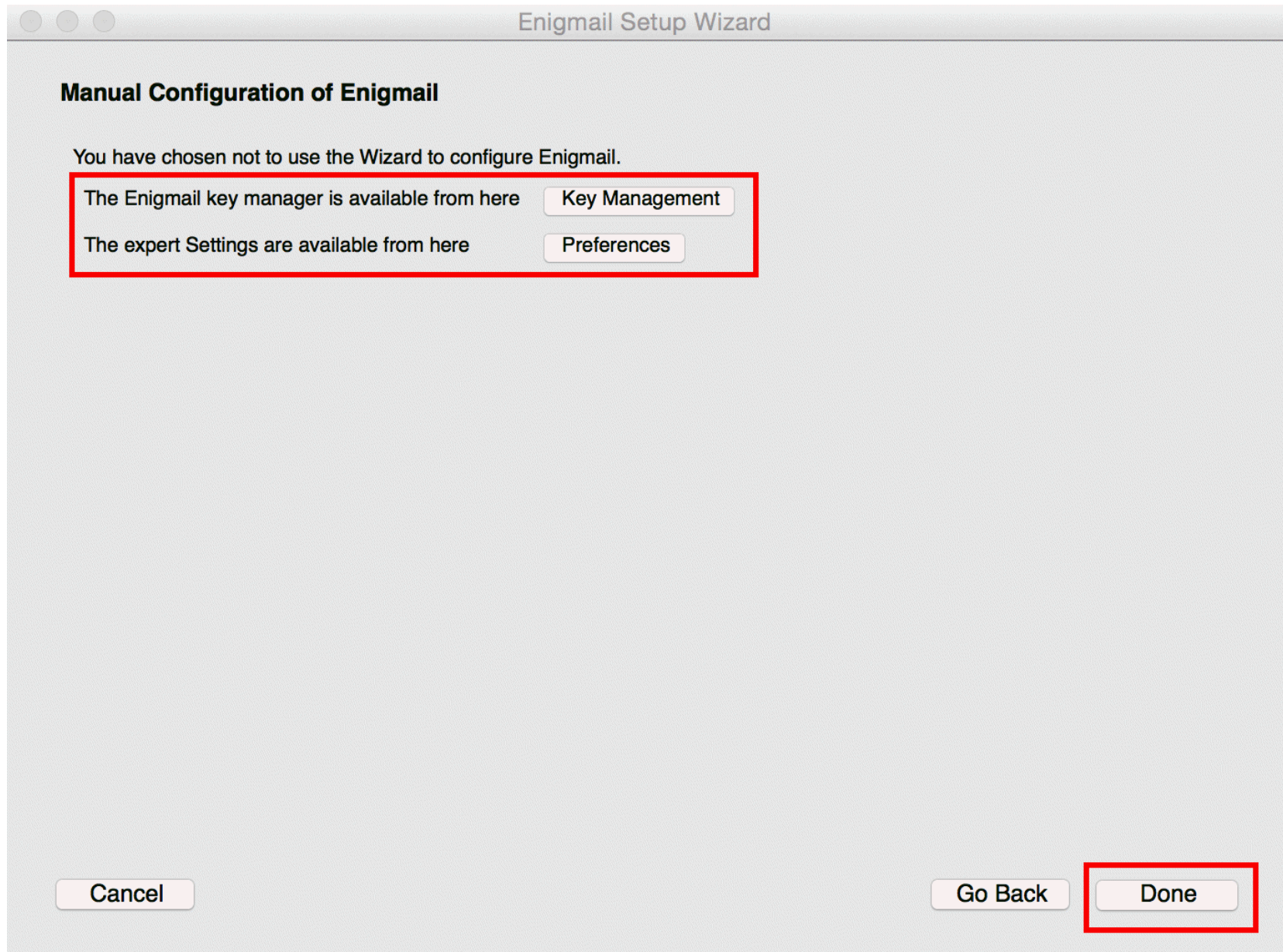
How would you like to Configure Enigmail?

Would you like to setup Enigmail manually, or do you need assistance in the setup process?

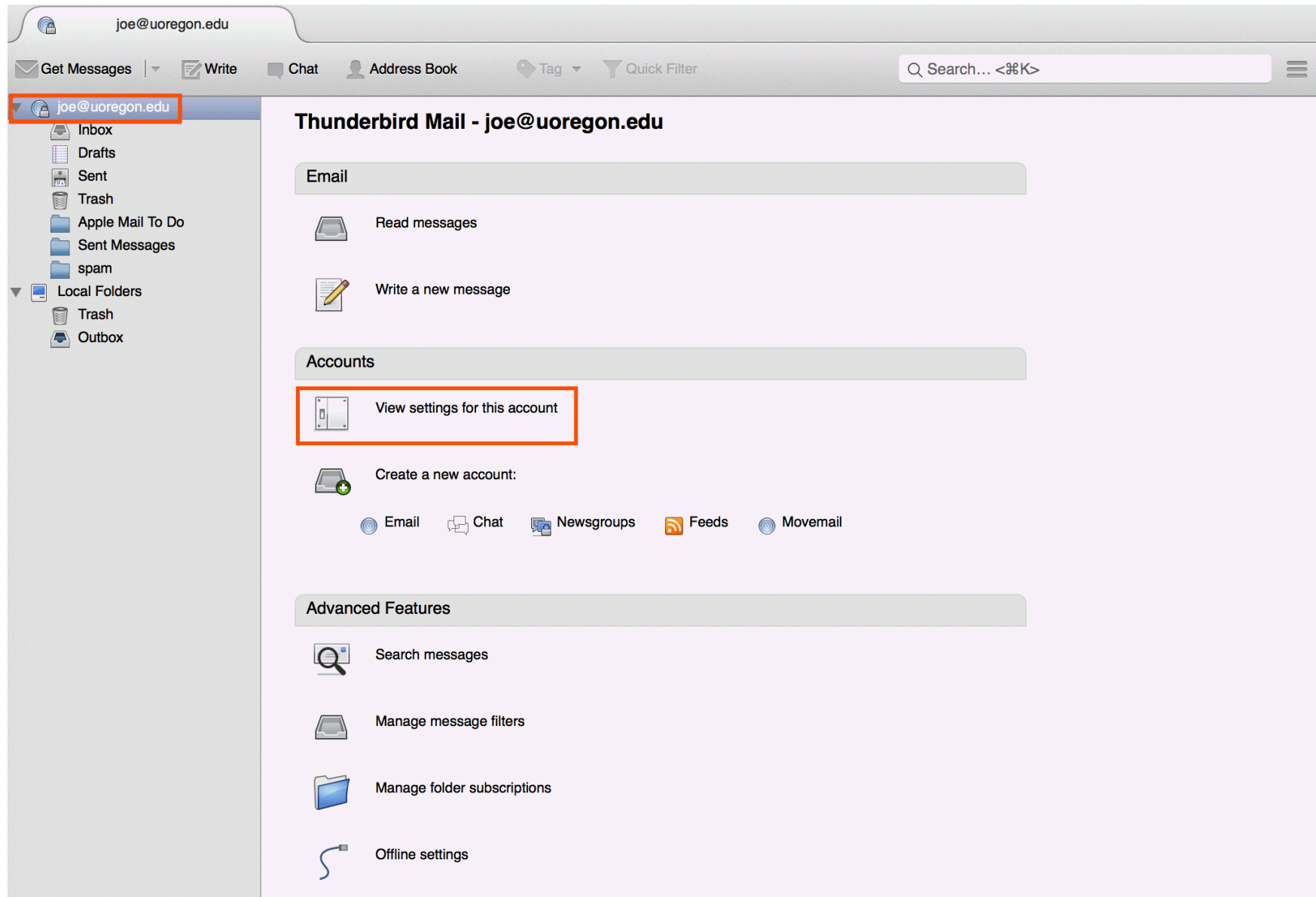
- ☐ I prefer a standard configuration (recommended for beginners)
- ☐ I prefer an extended configuration (recommended for advanced users)
- ☒ I prefer a manual configuration (recommended for experts)

The third option, "I prefer a manual configuration (recommended for experts)", is highlighted with a red rectangular border. At the bottom of the dialog, there are three buttons: "Cancel" on the left, "Go Back" in the center, and "Continue" on the right. The "Continue" button is also highlighted with a red rectangular border.

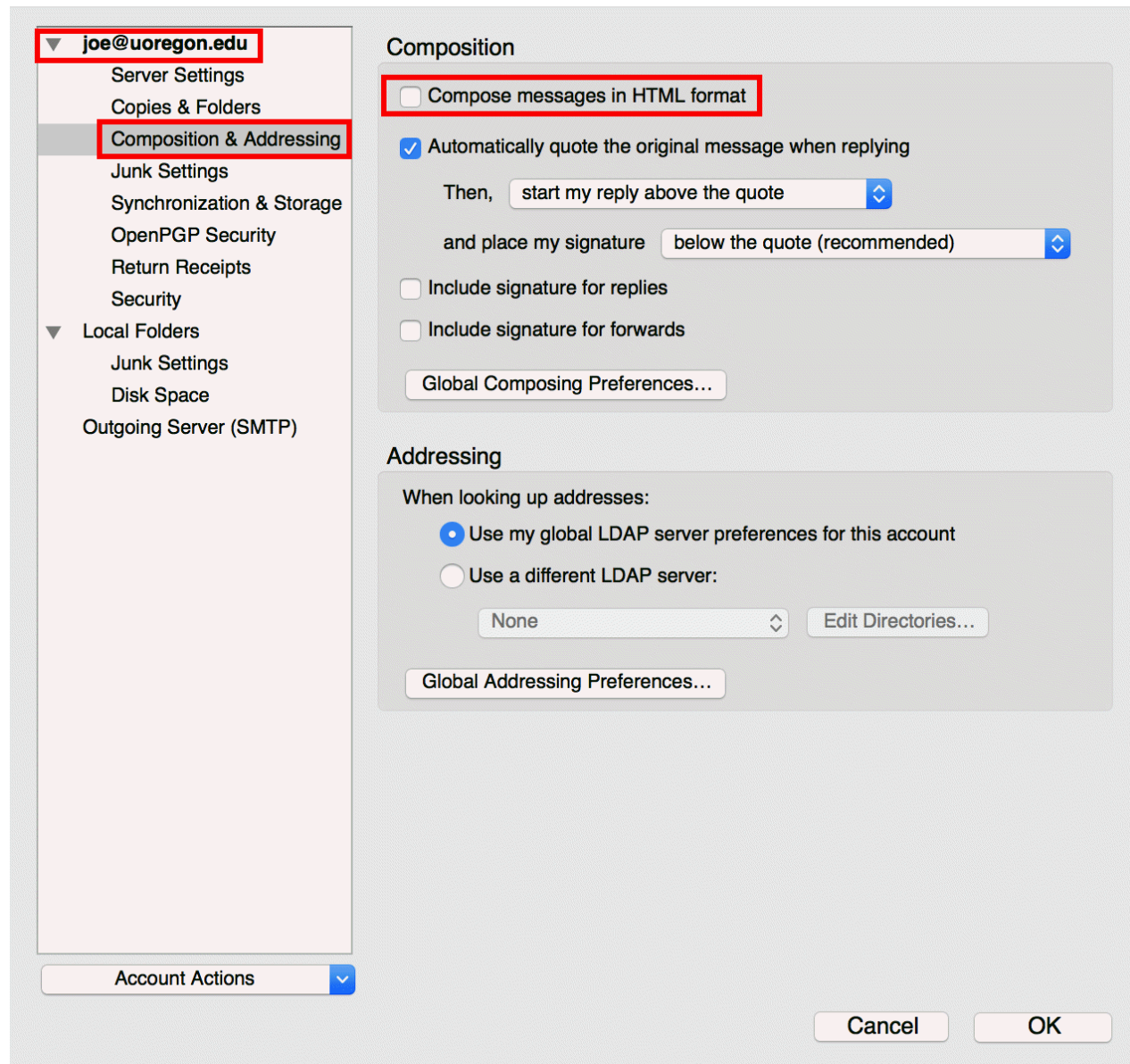
Basically Just Click Done



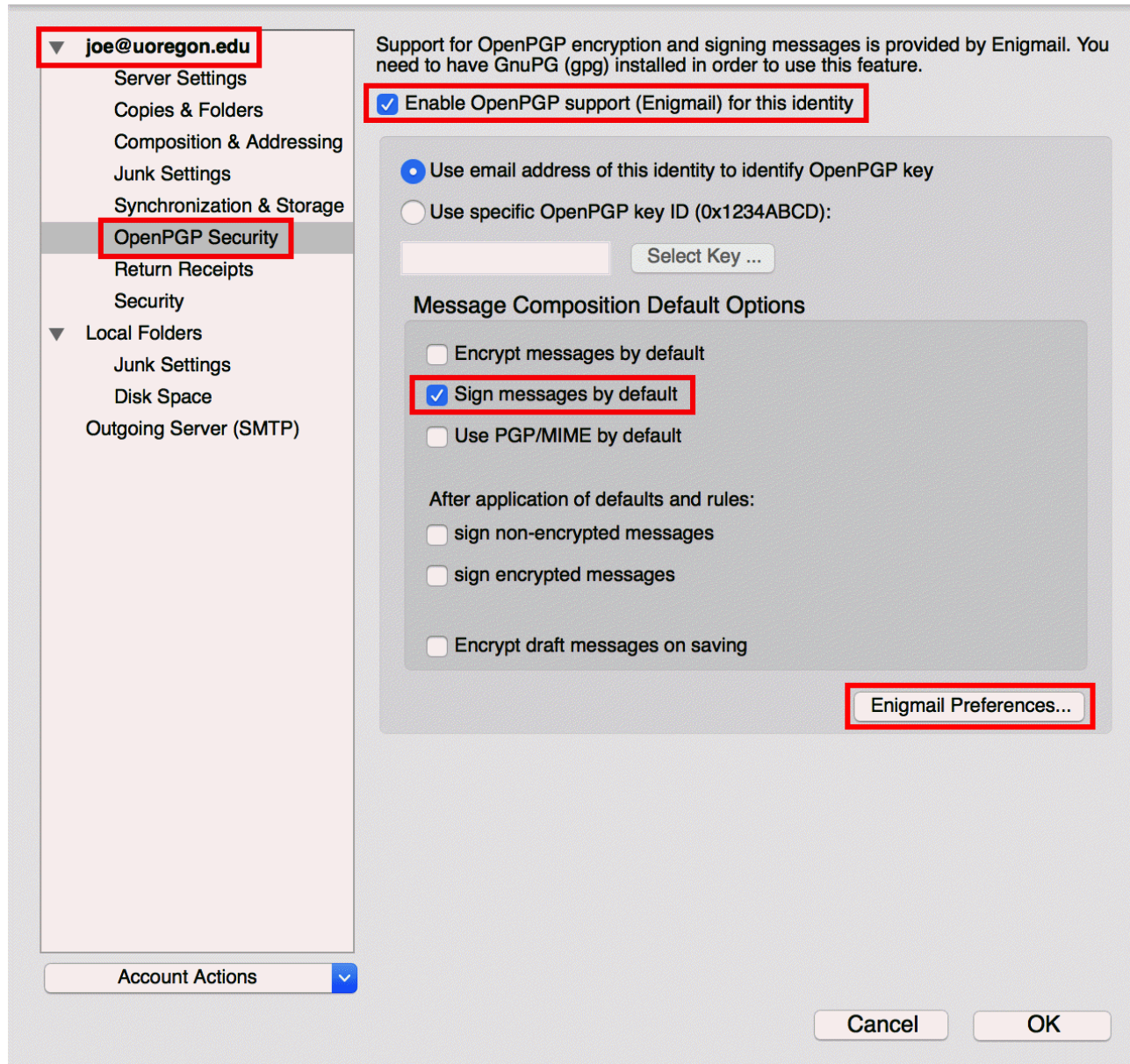
Manually Configuring Settings For Your Account



NO HTML! Compose in Plain Text ONLY



Configuring OpenPGP Security



Configuring Enigmail Preferences: Basic Is OK As-Is

The image shows a screenshot of the 'Basic Settings' dialog box in Enigmail. At the top, there are two tabs: 'Basic' (selected) and 'Sending'. The 'Basic Settings' section is divided into two main areas. The first area, 'Files and Directories', shows that 'GnuPG was found in /usr/local/bin/gpg'. Below this, there is a checkbox labeled 'Override with' followed by a text input field and a 'Browse...' button. The second area, 'Passphrase settings', shows 'Remember passphrase for' followed by a spinner box set to '5' and the text 'minutes of idle time'. Below this is a checkbox labeled 'Never ask for any passphrase'. At the bottom of the dialog, there is a button labeled 'Display Expert Settings and Menus' and a 'Reset' button. At the very bottom of the window are 'Cancel' and 'OK' buttons.

Basic Sending

Basic Settings

Files and Directories

GnuPG was found in /usr/local/bin/gpg

☐ Override with Browse...

Passphrase settings

Remember passphrase for minutes of idle time

☐ Never ask for any passphrase

Display Expert Settings and Menus

Reset

Cancel OK

Configuring Enigmail Preferences: Sending Tab Tweaks

Basic Sending

General Preferences for Sending

☐ Convenient encryption settings Help

☒ Manual encryption settings

☒ Encrypt/sign replies to encrypted/signed messages

Automatically send encrypted

☐ Never

☒ If possible

To send encrypted, accept

☒ Only trusted keys

☐ All usable keys

Confirm before sending

☐ Never

☒ Always

☐ If encrypted

☐ If unencrypted

☐ If rules changed the default encryption setting

See also account specific Enigmail settings (see Account Settings -> OpenPGP Security)

Cancel OK



Notes: Enigmail Can Be Quite Particular

- Enigmail "works best" if your GPG key matches the account you've got configured in Thunderbird, character for character.
- If you want to use an existing keypair (associated with a different userid) for an account you've got configured in Thunderbird, you can go to Enigmail → Key Management and add an additional identity to your existing key pair:
 - select the key that's going to get the additional identity
 - right click on that key
 - click on Manage User IDs
 - click on Add
 - provide the new identity
 - provide your passphrase for the existing ID
 - select one of the IDs to be "primary"

Write A Message With Thunderbird & Enigmail

Write: Test message from Enigmail

Send | ✓ Spelling | Attach | S/MIME | Save

Enigmail:   Attach My Public Key This message will be signed and encrypted

From: joe st sauver <joe@uoregon.edu> *joe@uoregon.edu*

To: *joe@stsauver.com*

Subject: Test message from Enigmail

Hi Joe,

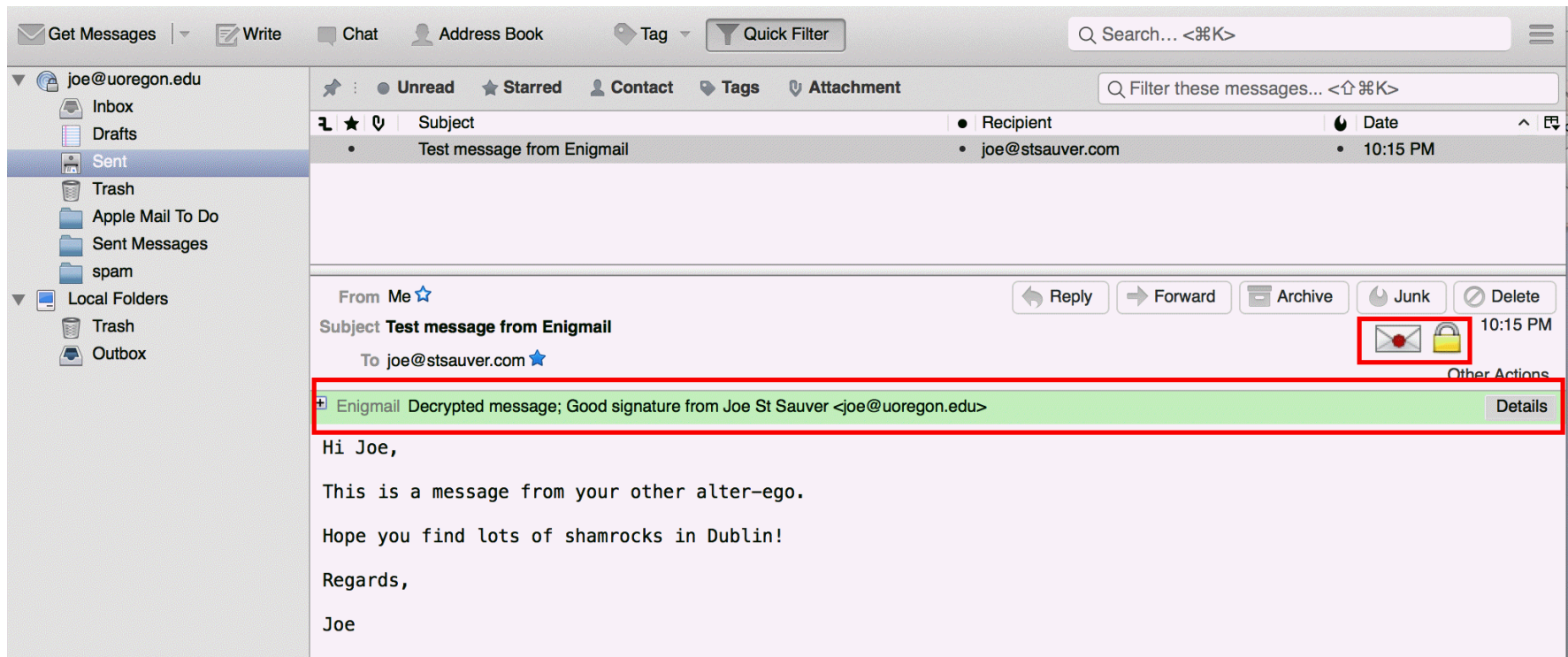
This is a message from your other alter-ego.

Hope you find lots of shamrocks in Dublin!

Regards,

Joe

Receive A Message Using Thunderbird & Enigmail



"But Joe! I Use A Webmail Account!"

- There are solutions for that circumstance, too.
- For example, many popular webmail account providers allow IMAP clients (such as Thunderbird) to connect to the "webmail" account. You don't HAVE to use the web email interface. You can use Thunderbird instead.
- However, if you like using a webmail account, as long as you're using your own laptop with Firefox or Chrome, you can use Mailvelope to add GPG support to most popular webmail interfaces.

See <https://www.mailvelope.com/> for more information

**KEY SIGNING
PARTY**

That's About All The Formal Presentation For Today

- I suspect you're full up to the gills at this point. That's okay. You've learned a lot today! We'll skip doing a summary here...
- By now you may have signed at least one class partner's account.
- If we have time, I'd like to offer the chance for other people to also arrange to have their keys signed.
- This is often done at a "PGP key signing party." See for example

http://www.cryptnet.net/fdp/crypto/keysigning_party/en/keysigning_party.html

OPTIONAL: IF You'd Like To Participate

- Upload your public key to the keyserver so others can easily find it
- Write your name, email address, and the fingerprint for your key on a slip of paper. If you want one other person to sign your key, make one such slip. If you're hoping to have six other people sign your key, make six such slips, etc. **PRINT *LEGIBLY*, PLEASE!**
- Ask a colleague to sign your key. **IF THEY'RE WILLING:**
 - give them one of your slips
 - show them your passport or driver's license
 - your colleague makes a notation that they've confirmed your identity on the slip they've received from you, *saving that slip*
- If you've agreed to sign someone's key, sometime in the next day or so, download a copy of their key, check to make sure the name and fingerprint match your slip for them, sign their key, and email it back to them. **PLEASE DO IT IF YOU SAY YOU'RE GOING TO!**

Thanks for Coming To Today's Training!

- Are there any questions?
- Please also remember to fill out the evaluations!
- Copies of this presentation are available online at

<https://www.stsauver.com/joe/pgp-tutorial/>

and may be freely shared with others who may be interested
(even outside of M3AAWG)